THE CIP NETWORKS LIBRARY

Volume 2

EtherNet/IP Adaptation of CIP

Edition 1.10

November 2010

The CIP Networks Library
Volume 2: EtherNet/IP Adaptation of CIP

Publication Number: PUB00002

ODVA, Inc.
4220 Varsity Drive, Suite A, Ann Arbor, MI 48108-5006 USA
TEL 1-734-975-8840
FAX 1-734-922-0027
EMAIL odva@odva.org
WEB www.odva.org

SITE SUBSCRIPTION

- The Final Specification, of which this volume and edition of The CIP Networks Library is a part, is provided on an annual subscription basis to this Licensed Vendor Member, as defined by its unique Vendor ID for the technology contained in the Final Specification ("YOU"), pursuant to your Terms of Usage Agreement with ODVA, Inc. (ODVA) for the technology contained in the Final Specification.
- This subscription is a site subscription, and this subscription, along with your membership in ODVA, must be renewed annually in order to maintain continuing rights to the site subscription as allowed under your Terms of Usage Agreement.
- This site subscription permits access to the electronic files for this volume contained on the distribution CD by multiple users who are your employees and on-site contracted individuals performing typical employee functions on a contract basis ("Authorized Users"). YOU shall ensure that, if access to these files if given to contractors, the contractors can fulfill the obligations of your Terms of Usage Agreement for the ODVA technology contained in the Final Specification. YOU shall not knowingly permit anyone other than Authorized Users to access these files.
- This site subscription permits access to the electronic files contained on the distribution CD for this volume via the original CD on which this volume is distributed or via an electronic copy of this volume placed on your single, secure intranet site. If and when the electronic files are posted on your intranet site, YOU shall relocate the original CD to secure storage for use only as a system back-up for the electronic files posted on your intranet site.
- The possession of or subscription to this Final Specification does not, by itself, convey any right to use or reproduce any portion of the Final Specification or to make, have made, use, import, offer to sell, sell, lease, market, or otherwise distribute or dispose of any products contemplated by the Final Specification, and you are hereby notified that the products contemplated by this Final Specification might be covered by valid patents or copyrights of ODVA, its members or other licensors. The necessary licenses to use or reproduce portions of the Final Specification for use in products, or to make, have made, use, import, offer to sell, sell, lease, market, or otherwise distribute and dispose of such products may be obtained only from ODVA through its Terms of Usage Agreement, available through the ODVA web site. This license requirement applies equally (a) to devices that completely implement the Final Specification with a network port that can be issued a Declaration of Conformity ("CIP Network Devices"), (b) to components of such CIP Network Devices to the extent they implement portions of the Final Specification, and (c) to enabling technology products, such as any CIP Network protocol stack, designed for use in CIP Network Devices to the extent they implement portions of the Final Specification. Contact ODVA for a Terms of Usage Agreement if you are not already licensed.
- Any other distribution and all other electronic copies, intranet or internet postings, and any printed copies are prohibited. Printed copies of this volume may be purchased via the order form available through the ODVA web site at www.odva.org.
- Notwithstanding anything to the contrary herein, if in connection with bookmarking a page of the Final Specification it is reasonably necessary for an Authorized User to possess a copy of the Final Specification on the computer on which such page is bookmarked, then you may possess such copy, provided that (i) such copy is not accessible to anyone other than the Authorized User, (ii) such copy is not retained for any longer than reasonably necessary to use the bookmarked page and in any event no longer than the term of this subscription, (iii) use of such copy is limited to the uses permitted in this site subscription and (iv) such copy is not transmitted or otherwise distributed to any other person, computer or device.

This page is intentionally left blank

The CIP Networks Library: Volume 3

EtherNet/IP Adaptation of CIP

Table of Contents

Revisions

The CIP Networks Library Volume 2: EtherNet/IP Adaptation of CIP Edition 1.10 contains the following changes from the previous Edition. Please see the change bars on the pages noted here for specific modifications. Note: Some of the pages within the ranges noted may not contain any changes.

Chapt-Sect	Pages	Description
		SockAddr Fix
3-3.9.5	3-13	• Remove the first paragraph that dealt with choosing the port number for point to point connections
		EtherNet/IP Quick Connect Functionality
1-5	1-8	• Add new definitions
5-3.2	5-5	• Update Revision history table
5-3.3.2	5-9	• Add new attribute 12 to instance attribute table
5-3.3.2.11	5-16	• Add subsection for EtherNet/IP Quick_Connect attribute
7-2.1	7-3	• Change Optional to Conditional for TCP/IP object and add footnote
7-6	7-6, 7	• Add new section/subsections for TCP/IP Interface object class
E	All	• Added new Appendix
		Common Services Cleanup
5-3.3.1	5-17	• Add footnote to Table 5-3.12
5-4.4.1	5-27, 28	• Changes to two paragraphs
		Ethernet Link Object Update
5-4.3.1	5-21	• Add NV column to Class Attributes table
5-4.3.2	5-22 thru 24	• Add NV column to Instance Attributes table
5-4.4.2	5-28	• Insert new table to show Get_Attributes_All response details
5-4.6	5-30	• Add Behavior section
		TCP/IP Object Cleanup and state diagram
5-3.2	5-5	• Inserted new heading for Revision History
5-3.3.1	5-6	• Add Class Attribute 1 and NV column to Class Attributes table
5-3.3.2	5-6 thru 5-9	• Add NV column, make changes to Access rule & footnotes in Table 5-3.3
5-3.3.2.1	5-10	• Add bit 5 definition to Status Attribute in Table 5-3.4
5-3.3.2.2	5-10	• Remove exclusion from bit 4 definition • Add bit 6 definition to Configuration Capability Attribute • Change bit 3 definition
5-3.3.2.3	5-10, 11	• Inserted subsection for Configuration Control Structure and applied edits to first paragraph and the table.
5-3.3.2.5.1 5-3.3.2.5.2 5-3.3.2.5.3	5-12 thru 5-14	• Inserted subsections and mostly new material
5-3.3.2.6	5-14	• Remove most of this section, replace with new material.
5-3.5	5-18 thru 5-20	• Delete "State Transition" from text in 1st paragraph and change the diagram
		Corrections related to Announce-based Node Operation
9-5.5.2.7	9-20	• Changes to 2nd paragraph to differentiate Announce and Beacon operation
9-5.9.2	9-42 9-44	• In Events 9 and 26 add action step b, Update VLAN ID (if different)
9-5.9.3	9-46	• Changes to 4th bullet to differentiate Announce and Beacon operation

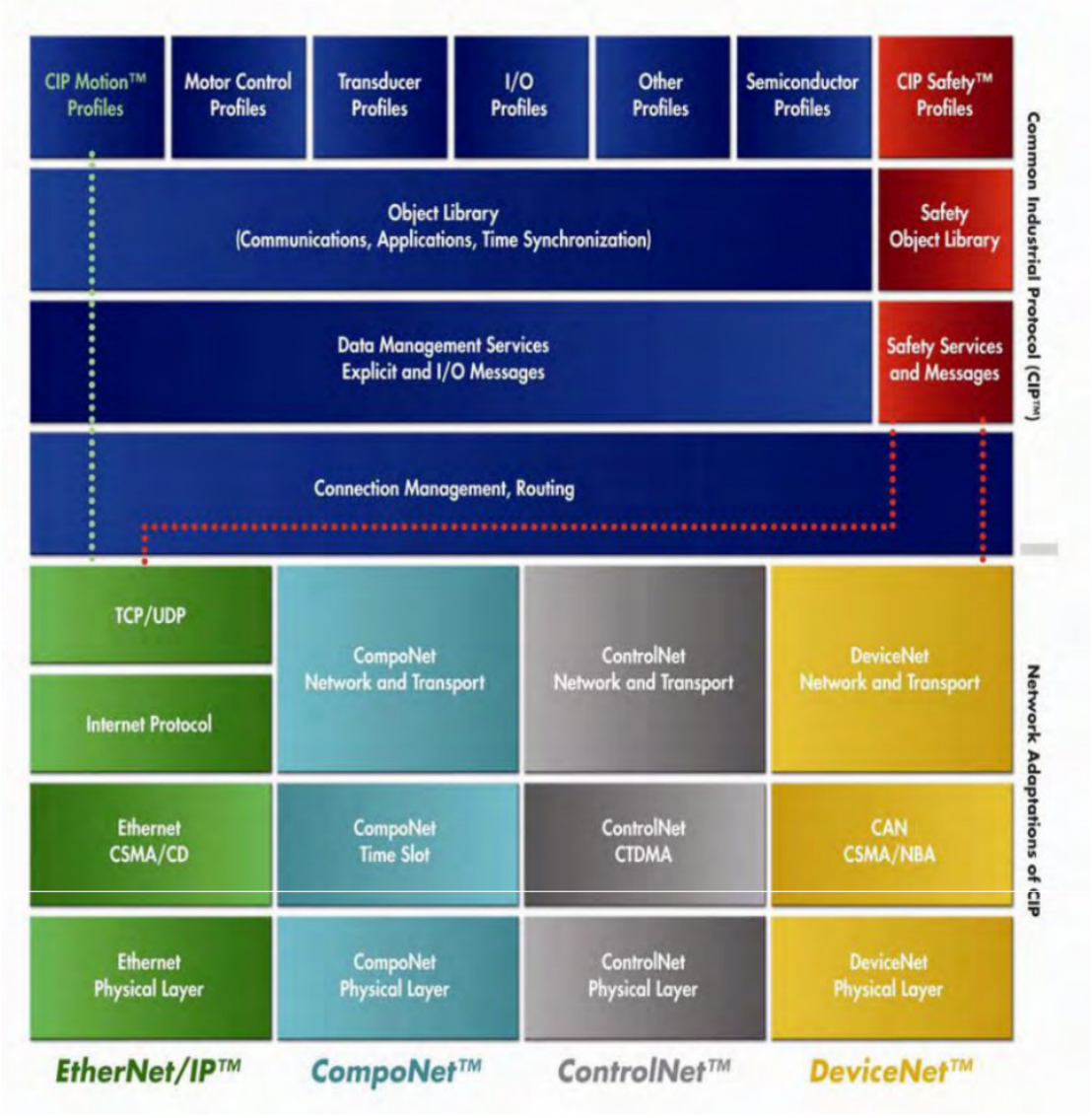
Chapt-Sect	Pages	Description
		DLR Frame Format Clarifications
9-5.8.1	9-30	• Modify item 2 in list
9-5.8.2	9-31	• Change last two rows of Table 9-5.4
9-5.8.3	9-31	• Add byte 40 remark to Table 9-5.5
9-5.8.4	9-32	• Add byte 30 remark to Table 9-5.7
9-5.8.5	9-31	• Add byte 31 remark to Table 9-5.8
9-5.8.6	9-32, 33	• Modify paragraph and change remarks for byte 30 & 31 in table 9-5.9 • Replace Table 9-5.10 with new material
9-5.8.7	9-33	• Add byte 30 remark to Table 9-5.11
9-5.8.8	9-34	• Add byte 31 remark to Table 9-5.12
9-5.8.9	9-34	• Change … row to indicate it is Reserved, show the data type and value
		IPv4 Address Conflict Detection (ACD)
5-3.2	5-5	• Updated Revision History
5-3.3.2	5-9	• Add Instance Attributes 10, 11
5-3.3.2.1	5-10, 11	• Add bit 6 definition to Status Attribute in Table 5-3.4
5-3.3.2.2	5-11	• Add bit 7 definition to Configuration Capability Attribute in Table 5-3.5
5-3.3.2.9 5-3.3.2.10	5-15, 16	• Add subsections for semantics of new attributes
F	All	• Add new Appendix
		Misc
2-6.3.3	2-35	• Indicate that sin_port is sent big endian. Previous enhancement not applied properly in a previous edition

Preface

Organization of the CIP Networks Specifications

Today, four networks - DeviceNet™, ControlNet™, EtherNet/IP™ and CompoNet™ - use the Common Industrial Protocol (CIP) for the upper layers of their network protocol. For this reason, ODVA manages and distributes the specifications for CIP Networks in a common structure to help ensure consistency and accuracy in the management of these specifications.

The following diagram illustrates the organization of the library of CIP Network specifications. In addition to CIP Networks, CIP Safety™ consists of the extensions to CIP for functional safety.



This common structure presents CIP in one volume with a separate volume for each network adaptation of CIP. The specifications for the CIP Networks are two-volume sets, paired as shown below.

The EtherNet/IP specification consists of:

Volume 1: Common Industrial Protocol (CIP™)

Volume 2: EtherNet/IP Adaptation of CIP

The DeviceNet specification consists of:

Volume 1: Common Industrial Protocol (CIP™)

Volume 3: DeviceNet Adaptation of CIP

The ControlNet specification consists of:

Volume 1: Common Industrial Protocol (CIP™)

Volume 4: ControlNet Adaptation of CIP

The CompoNet specification consists of:

Volume 1: Common Industrial Protocol (CIP™)

Volume 6: CompoNet Adaptation of CIP

The specifications for CIP Safety™ is distributed in a single volume:

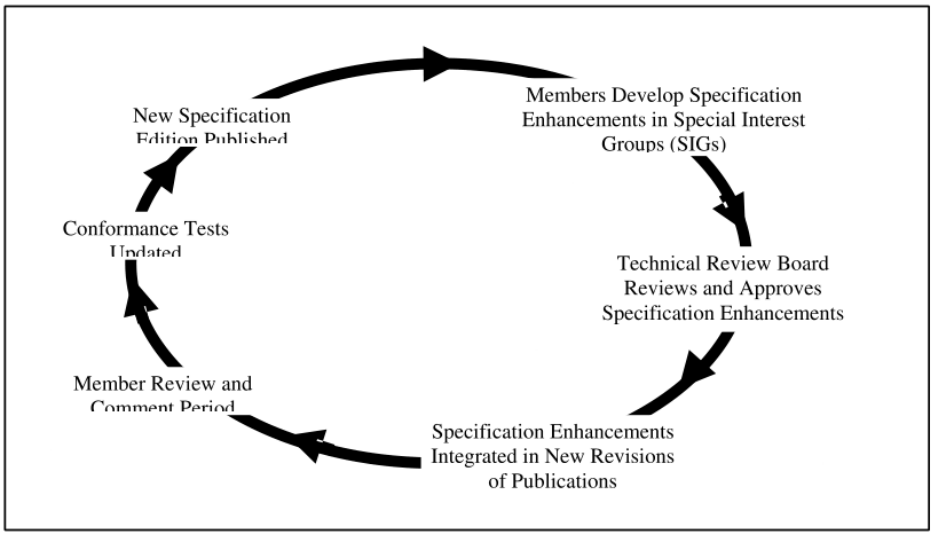
Volume 5: CIP Safety

The specification for integrating Modbus Devices is distributed in a single volume:

Volume 7: Integration of Modbus Devices into the CIP Architecture

Specification Enhancement Process

The specifications for CIP Networks are continually being enhanced to meet the increasing needs of users for features and functionality. ODVA has implemented a Specification Enhancement Process in order to ensure open and stable specifications for all CIP Networks. This process is ongoing throughout the year for each CIP Network Specification as shown in the figure below. New editions of each CIP Network specification are published on a periodic basis.



This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 1: Introduction to EtherNet/IP

Contents

1-1 Introduction

EtherNet/IP (Ethernet/Industrial Protocol) is a communication system suitable for use in industrial environments. EtherNet/IP allows industrial devices to exchange time-critical application information. These devices include simple I/O devices such as sensors/actuators, as well as complex control devices such as robots, programmable logic controllers, welders, and process controllers.

EtherNet/IP uses CIP (Common Industrial Protocol), the common network, transport and application layers also shared by ControlNet and DeviceNet. EtherNet/IP then makes use of standard Ethernet and TCP/IP technology to transport CIP communications packets. The result is a common, open application layer on top of open and highly popular Ethernet and TCP/IP protocols.

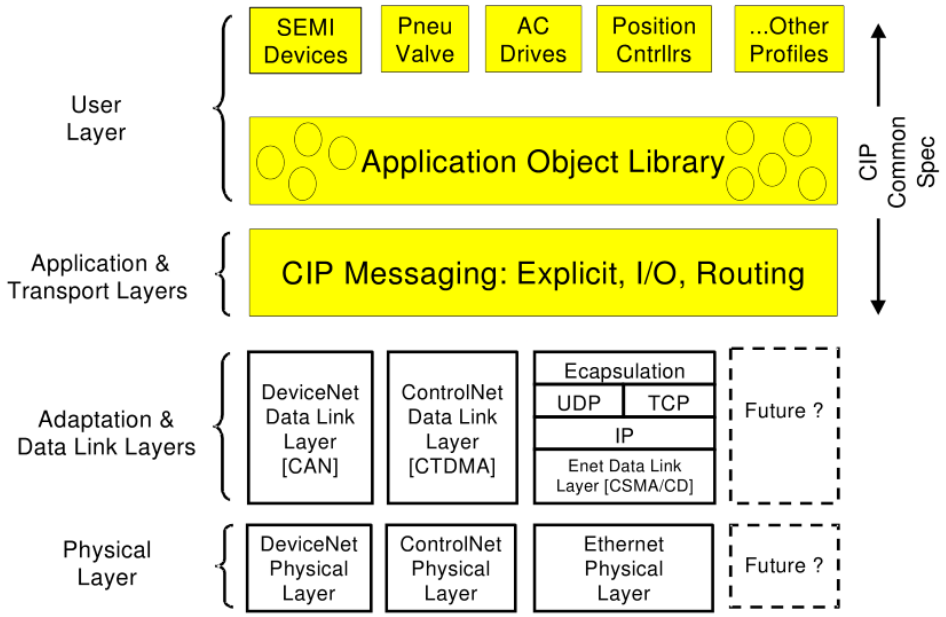
EtherNet/IP provides a producer/consumer model for the exchange of time-critical control data. The producer/consumer model allows the exchange of application information between a sending device (e.g., the producer) and many receiving devices (e.g., the consumers) without the need to send the data multiple times to multiple destinations. For EtherNet/IP, this is accomplished by making use of the CIP network and transport layers along with IP Multicast technology. Many EtherNet/IP devices can receive the same produced piece of application information from a single producing device.

EtherNet/IP makes use of standard IEEE 802.3 technology; there are no non-standard additions that attempt to improve determinism. Rather, EtherNet/IP recommends the use of commercial switch technology, with 100 Mbps bandwidth and full-duplex operation, to provide for more deterministic performance.

NOTE: EtherNet/IP does not require specific implementation or performance requirements due to the broad range of application requirements. However, work is underway to define a standard set of EtherNet/IP benchmarks and metrics by which the performance of devices will be measured. These measurements may become required entries within a product's Electronic Data Sheet. The goal of such benchmarks and metrics will be to help the user determine the suitability of a particular EtherNet/IP device for a specific application.

The figure below illustrates how EtherNet/IP, DeviceNet and ControlNet share the CIP Common layers.

Figure 1-1.1 CIP Common Overview



1-2 Scope

The EtherNet/IP specification is divided into the following chapters:

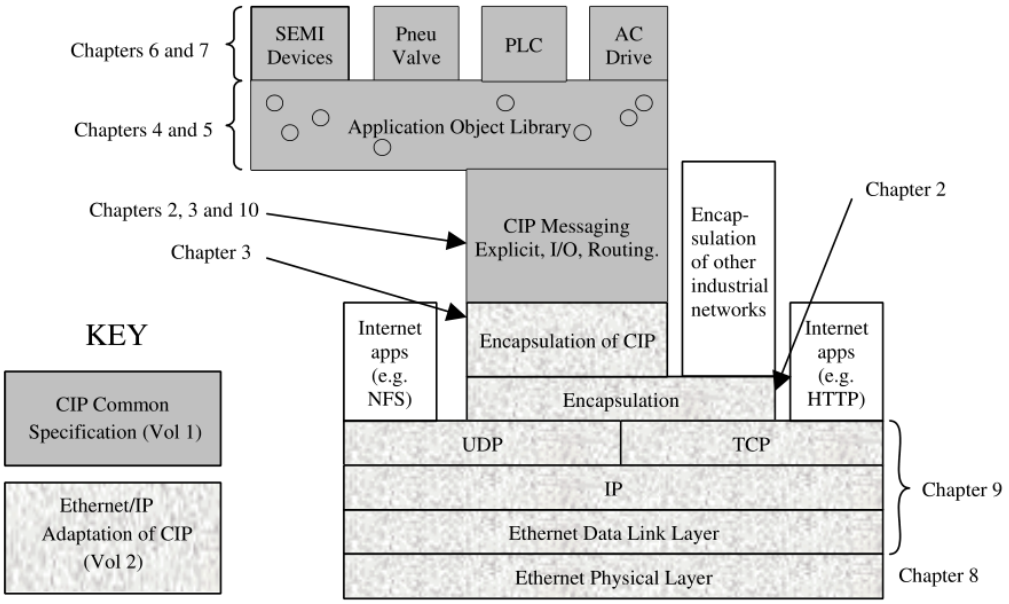
Chapter	Title	Description
1	Introduction	This chapter of the specification.
2	Encapsulation Protocol	Specifies the encapsulation protocol that is used to transport CIP packets over TCP/IP networks. The encapsulation protocol specified in this chapter may also be used to encapsulate non-CIP protocols.
3	Mapping of Explicit and I/O Messaging to TCP/IP	Contains EtherNet/IP-specific additions to the CIP Network and Transport layers. Specifies how the encapsulation protocol defined in Chapter 2 is used to transport CIP Network and Transport layer packets over TCP/IP networks.
4	Object Model	Contains EtherNet/IP-specific additions to the CIP object model.
5	Object Library	Supplements the CIP object library with objects specific to EtherNet/IP.
6	Device Profiles	Contains EtherNet/IP-specific additions to the CIP device profile library.
7	Electronic Data Sheets	Specifies additions to the CIP EDS definition required for EtherNet/IP.
8	Physical Layer	Specifies media and physical layer requirements for industrial use.
9	Indicators and Middle Layers	Specifies TCP/IP requirements of EtherNet/IP devices. This chapter also specifies the standard appearance and behavior of EtherNet/IP diagnostic LEDs.
10	Bridging and Routing	Additions to the CIP routing definition.

This chapter is the Introduction to EtherNet/IP. The following drawing shows the relationship of these chapters to each other and to the CIP Common specification (published separately by ODVA and ControlNet International). Both this specification (volume2) and the CIP Common specification (volume1) are required to completely specify an EtherNet/IP product. The encapsulation protocol defined in Chapter 2 of this specification is also suitable to encapsulate other industrial protocols, as illustrated in the following drawing. However, the specific details of encapsulating other protocols are not included in this release of the specification.

As can be seen in Figure 1-2.1, the encapsulation protocol in chapter 2 uses a TCP/IP layer to insulate it from the network medium. As such, the encapsulation protocol may be used on any medium that supports TCP/IP. For example, the encapsulation protocol could run on an FDDI or PPP network. Chapter 9 (Indicators and Middle Layers) requires conformance with the RFC that documents how TCP/IP is implemented on a particular network. Furthermore, chapter 8 (Physical Layers) narrows the scope of certified EtherNet/IP implementations to run on either 10 or 100 Mb Ethernet. Specifically, chapter documents two permissible conformance levels of devices: one called "commercial" and the other "industrial". Other conformance levels may be added through modification to this specification.

Figure 1-2.1 shows the relationship between the various parts of the EtherNet/IP specification. As shown in the figure, the darker sections (chapters 2-7 and 10) are predominately documented by the CIP Common specification (volume 1). The corresponding chapters of the EtherNet/IP Adaptation of CIP (volume 2) supplements or modifies these chapters of the CIP Common specification in some areas. The lightly shaded sections (chapters 2, 3, 8 and 9) are predominately documented by volume 2. These chapters contain information applicable specifically to EtherNet/IP devices, but not necessarily to those on other CIP networks (for example, DeviceNet or ControlNet).

Figure 1-2.1 Document Organization Overview



1-3 References

1-3.1 Normative References

ISO 7498-1:1984, Information processing systems — Open systems interconnection — Basic reference model

ISO 7498/AD1: 1987, Information processing systems — Open systems interconnection — Connectionless data transmission

ISO 7498-3:1987, Information processing systems — Open systems interconnection — Naming and addressing

ISO/IEC 8886:1992, Information technology — Open systems interconnection — Telecommunications and information exchange between systems — Data link service definition

ISO/IEC 10039:1990, Information technology — Telecommunication and information exchange between systems — Medium access control service definition

ISO/TR 8509:1987, Information processing systems — Open systems interconnection — Service conventions

ISO/IEC 10731:1992, Information technology — Open systems interconnection — Conventions for the definition of OSI services

ISO 8802-2:1989, Information processing systems — Local area networks — Part 2: Logical link control

ISO/IEC 8802-3:1993, Information technology — Local and metropolitan area networks — Part 3: Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications

ISO/IEC 8802-4:1990, Information processing systems — Local area networks — Part 4: Token - passing bus access method and physical layer specifications

ANSI X3.159-1989, American National Standard for Information Systems — Programming Language C

1-4 Additional Reference Material

"Strategies for Real-time Systems Specification" by D. J. Hatley and I. A. Pirbhai

CEN/CENELEC Internal Regulations Part 3: Rules for the drafting and presentation of European Standards (PNE-Rules) - 1991-09

RFC 768: August 1980, User Datagram Protocol

RFC 791: September 1981, Internet Protocol

RFC 792: September 1981, Internet Control Message Protocol

RFC 793: September 1981, Transmission Control Protocol

RFC 826: November 1982, An Ethernet Address Resolution Protocol

RFC 894: April 1984, A Standard for the Transmission of IP Datagrams over Ethernet Networks

RFC 1035:1987, Domain names - implementation and specification

RFC 1103: June 1989, A Proposed Standard for the Transmission of IP Datagrams over FDDI Networks

RFC 1112: August 1989, Host Extensions for IP Multicasting

RFC 1117:1989, Internet numbers

RFC 1122: October 1989, Requirements for Internet Hosts -- Communication Layers

RFC 1123: October 1989, Requirements for Internet Hosts -- Application and Support

RFC 1127: October 1989, A Perspective on the Host Requirements RFCs

RFC 1171: July 1990, The Point-to-Point Protocol for the Transmission of Multi-Protocol Datagrams Over Point-to-Point Links

RFC 1201: February 1991, Transmitting IP Traffic over ARCNET Networks

RFC 1392: January 1993, Internet Users' Glossary

RFC2236: November 1997, Internet Group Management Protocol, Version 2

1-5 Definitions

For the purposes of this standard, the following definitions apply. Also see CIP Common Specification, Chapter 1 for additional definitions.

Term	Definition
automation outlet	The interface where the generic telecommunications cabling ends and the automation specific cabling begins, including the interfaces where automation specific cabling terminates within the automation island.
broadcast	A special type of multicast packet that all nodes on the network are always willing to receive. [Source: RFC1392]
broadcast storm	An incorrect packet broadcast onto a network that causes multiple hosts to respond all at once, typically with equally incorrect packets which causes the storm to grow exponentially in severity. [Source: RFC1392]
datagram	A self-contained, independent entity of data carrying sufficient information to be routed from the source to the destination computer without reliance on earlier exchanges between this source and destination computer and the transporting network. [Source: RFC1392]
bulkhead	A wall or barrier which maintains the ingress and climatic environmental classification applicable on either side
bulkhead connection	An assembly of two back-to-back connections separated by a bulkhead
bulkhead cable gland	A device at an enclosure bulkhead that provides cable passage for power or signals
channel	A channel is defined as the end-to-end transmission path between two points at which application-specific equipment is connected. Alternatively a channel is a path of data transfer between two end devices
Controller	CIP connection originator, typically a robot controller or programmable controller
Quick Connect	A device mode that allows an EtherNet/IP target device to power up and be ready to accept a TCP connection in less than 350ms (for Class A Quick Connect targets).
encapsulation	The technique used by layered protocols in which a layer adds header information to the protocol data unit (PDU) from the layer above. As an example, in Internet terminology, a packet would contain a header from the physical layer, followed by a header from the network layer (IP), followed by a header from the transport layer (TCP), followed by the application protocol data. [Source: RFC1208]
Ethernet	A 10-Mb/s standard for LANs, initially developed by Xerox, and later refined by Digital, Intel and Xerox (DIX). All hosts are connected to a coaxial cable where they contend for network access using a Carrier Sense Multiple Access with Collision Detection (CSMA/CD) paradigm. See also: 802.x, Local Area Network, token ring. [Source: RFC1392]
EtherNet/IP	Products compliant with this specification as well as the CIP Common specification are known as EtherNet/IP products. EtherNet/IP stands for Ethernet Industrial Protocol. [Source: RFC1392]
frame	Single data transfer on a link.
link	The physical connection between two active mating components
MAC ID	the 48-bit physical address of an Ethernet node
network status indicators	Indicators on a node indicating the status of the Physical and Data Link Layers.
network address or node address	A node's 32-bit TCP/IP address on the link. In most CIP networks, this network address is the MAC ID; however, this is not the case on Ethernet. The DLL of Ethernet has a 48-bit MAC ID that is not used directly by the CIP communication stack.

Term	Definition
Numerical Aperture	In optics, the numerical aperture (NA) of an optical system is a dimensionless number that characterizes the range of angles over which the system can accept or emit light. The exact definition of the term varies slightly between different areas of optics
physical medium dependant	An active interface defined by the appropriate standards to serve a specific medium such as copper 2/4 pair or fiber
physical topology	The physical layout of devices on a network, or the way that the devices on a network are arranged and how they communicate with each other, is called the physical topology
port	Within the EtherNet/IP specific context, a TCP or UDP port is a transport layer demultiplexing value. Each application has a unique port number associated with it. [Source: RFC1392]. See CIP Common Specification for an additional definition of this term.
Power over Ethernet	The delivery of device power along with Ethernet signals, as defined by 803.3an in cooperation with TIA-TR42 standards committee
redundant media	A system using more than one medium to help prevent communication failures.
segment	Trunk–cable sections connected via taps with terminators at each end; a segment has no active components and does not include repeaters.
transceiver	The physical component within a node that provides transmission and reception of signals onto and off of the medium.

1-6 Abbreviations

For the purposes of this standard, the following abbreviations apply. Also see the CIP Common Specification Chapter 1 for additional abbreviations.

Abbreviation	Meaning
ACD	Address Conflict Detection
AO	Automation Outlet
COTS	Commercially off the shelf. Refers to commercial grade components
FTP	File transfer protocol. An internet application that uses TCP reliable packet transfer to move file between different nodes. (not to be confused with STP/FTP)
LED	Light emitting diode
NA	Numerical Aperture
PMD	Physical Media Dependant
PoE	Power over Ethernet
POF	Polymer Optical Fiber, formerly Plastic Optical Fiber
rcv	Receive
RFC	Request For Comments (RFC) – The document series, begun in 1969, which describes the Internet suite of protocols and related experiments. Not all (in fact, very few) RFCs describe the Internet standards, but all Internet standards are written up as RFCs. The RFC series of documents is unusual in that the proposed protocols are forwarded by the Internet research and development community, acting on their own behalf, as opposed to the formally reviewed and standardized protocols that are promoted by organizations such as CCITT and ANSI. [Source: RFC 1392]
rx	Receive
STP/FTP	Shielded twisted pair/foil twisted pair
TCP	Transmission Control Protocol (TCP) - An Internet Standard transport layer protocol defined in STD 7, RFC 793. It is connection-oriented and stream-oriented, as opposed to UDP. See also: connection-oriented, stream-oriented, User Datagram Protocol. [Source: RFC1392]
Tx	transmit
UDP	User Datagram Protocol (UDP) - An Internet Standard transport layer protocol defined in STD 6, RFC 768. It is a connectionless protocol which adds a level of reliability and multiplexing to IP. See also: connectionless, Transmission Control Protocol. [Source: RFC1392]
UTP	Unshielded twisted pair
Xmit	transmit

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 2: Encapsulation Protocol

Contents

2-1 Introduction

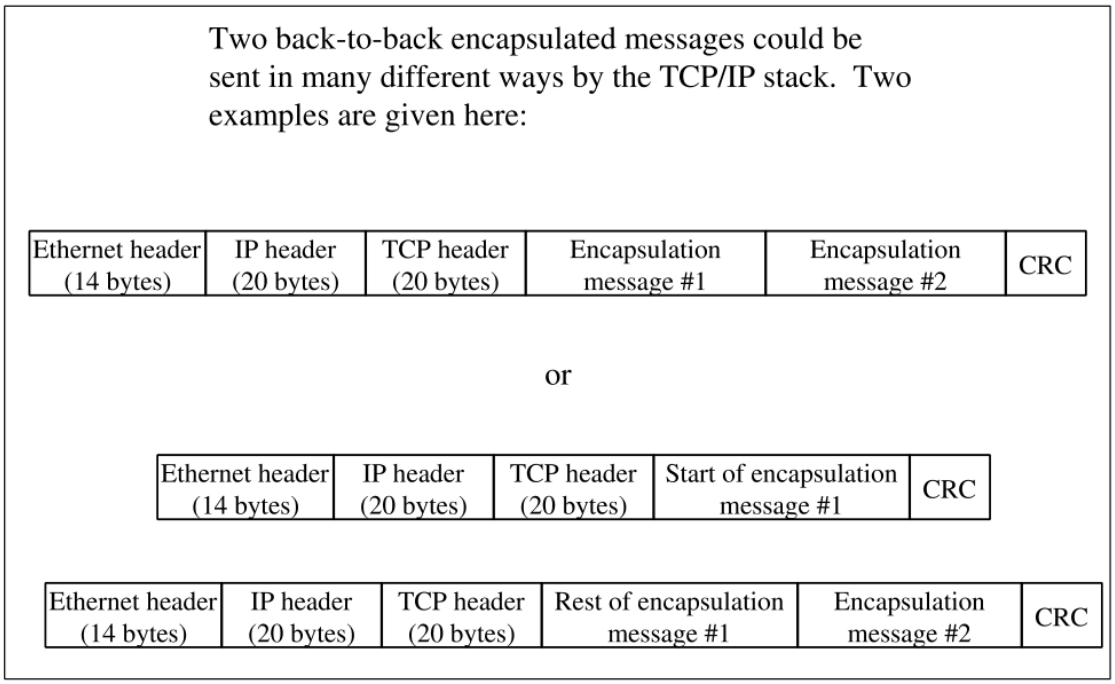
This chapter (chapter 2) of the specification documents the method used to encapsulate industrial protocols on a TCP/IP network. This mechanism can be applied to the CIP industrial protocol or to other networks. Chapter 3 of this specification details the application of this encapsulation protocol to CIP.

2-2 Use of TCP and UDP

The encapsulation protocol defines a reserved TCP port number that shall be supported by all EtherNet/IP devices. All EtherNet/IP devices shall accept at least 2 TCP connections on TCP port number 0xAF12. Once the TCP connection to TCP port number 0xAF12 is established, all data sent through the TCP stream shall be in the format specified in section 2-3.

NOTE: TCP is a stream-based protocol. It is permitted to send almost any length IP packet it chooses. For example, if two back-to-back encapsulated messages were passed to a TCP/IP stack, the TCP/IP stack may choose to put both encapsulated messages in one Ethernet frame. Alternately, it may choose to place half of the first message in the first Ethernet frame and all the rest in the next Ethernet frame. This is shown in Figure 2-2.1.

Figure 2-2.1 Usage of TCP to Encapsulate Two Messages



NOTE: It is not the intention of this specification to document the details of the TCP, UDP and IP transport mechanisms. Many excellent resources including the RFCs referenced throughout this specification should be used to obtain this information.

The encapsulation protocol also defines a reserved UDP port number that shall be supported by all EtherNet/IP devices. All devices shall accept UDP packets on UDP port number 0xAF12. Whenever UDP is used to send an encapsulated message, the entire message shall be sent in a single UDP packet. Only one encapsulated message shall be present in a single UDP packet destined to UDP port 0xAF12.

Some encapsulated messages shall only be sent via TCP. Other may be sent via either UDP or TCP. See "Table 2-3.2 Encapsulation Commands" for details about which commands are restricted to TCP.

2-3 Encapsulation Messages

2-3.1 Encapsulation Packet Structure

All encapsulation messages, sent via TCP or sent to UDP port 0xAF12, shall be composed of a fixed-length header of 24 bytes followed by an optional data portion. The total encapsulation message length (including header) shall be limited to 65535 bytes. Its structure shall be as follows:

Table 2-3.1 Encapsulation Packet

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	Encapsulation command
	Length	UINT	Length, in bytes, of the command specific data portion of the message, i.e., the number of bytes following the header
	Session handle	UDINT	Session identification (application dependent)
	Status	UDINT	Status code
	Sender Context	ARRAY of octet	Information pertinent only to the sender of an encapsulation command. Length of 8.
	Options	UDINT	Options flags
Command specific data	Encapsulated data	ARRAY of 0 to 65511 octet	The encapsulation data portion of the message is required only for certain commands

The encapsulation message length shall not override length restrictions imposed by the encapsulated protocol.

Multi-byte integer fields in the encapsulation messages shall be transmitted in little-endian byte order.

NOTE: This is different from the byte ordering used in standard Internet network protocols, which is big-endian.

Although the header contains no explicit information to distinguish between a request and a reply, this information shall be determined in either of two ways:

- implicitly, by the command and the context in which the message is generated. (For example, in the case of the RegisterSession command, the request is generated by an originator and the target generates the reply);
- explicitly, by the contents of an encapsulated protocol packet in the data part of the message.

2-3.2 Command Field

The allocation of command codes shall be as follows:

Table 2-3.2 Encapsulation Commands

Code	Name	Comment
0x0000	NOP	(may be sent only using TCP)
0x0001	Reserved for legacy usage [1]	
0x0002 and 0x0003	Reserved for legacy usage [1]	
0x0004	ListServices	(may be sent using either UDP or TCP)
0x0005	Reserved for legacy usage [1]	
0x0006 through 0x0062	Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)	
0x0063	ListIdentity	(may be sent using either UDP or TCP)
0x0064	ListInterfaces	**optional** (may be sent using either UDP or TCP)
0x0065	RegisterSession	(may be sent only using TCP)
0x0066	UnRegisterSession	(may be sent only using TCP)
0x0067 through 0x006E	Reserved for legacy usage [1]	
0x006F	SendRRData	(may be sent only using TCP)
0x0070	SendUnitData	(may be sent only using TCP)
0x0071	Reserved for legacy usage [1]	
0x0072	IndicateStatus	**optional** (may be sent only using TCP)
0x0073	Cancel	**optional** (may be sent only using TCP)
0x0074 through 0x00C7	Reserved for legacy usage [1]	
0x00C8 through 0xFFFF	Reserved for future expansion of this specification (Products compliant with this specification shall not used command codes in this range)	

1 Commands marked as “Reserved for legacy usage” indicate commands that were defined prior to the publication of this specification. Their behavior is undefined in this specification. Devices should not implement these commands without prior knowledge of the legacy usage. Devices that do not support these commands shall return encapsulation status code 0x0001.

A device shall accept commands that it does not support without breaking the session or underlying TCP connection. A status code indicating that an unsupported command was received shall be returned to the sender of the message.

NOTE: The establishment of a session is defined in section 2-5. In short, a session makes a TCP/IP connection between originator and target over which encapsulated commands may be sent. Since TCP/IP connections are modeled as a stream of bytes, the encapsulation header is prepended to each encapsulated packet so that the receiving device can know where packets begin and end.

2-3.3 Length Field

The length field in the header shall specify the size in bytes of the data portion of the message. The field shall contain zero for messages that contain no data. The total length of a message shall be the sum of the number contained in the length field plus the 24-byte size of the encapsulation header.

The entire encapsulation message shall be read from the TCP/IP connection even if the length is invalid for a particular command or exceeds the host's internal buffers. Data that would exceed internal buffers may be discarded, however the entire encapsulation messages must be read. Failure to read the entire message can result in losing track of the message boundaries in the TCP byte stream.

2-3.4 Session Handle

The Session Handle shall be generated by the target and returned to the originator in response to a RegisterSession request. The originator shall insert it in all subsequent encapsulation commands (sent using the commands listed in Table 2-3.2) which require sessions to that particular target. In the case where the target initiates and sends a command to the originator, the target shall include this field in the request that it sends to the originator.

Some commands (e.g., NOP) do not require a session handle even if a session has been established. The description of a particular command will note if it does not require a session.

2-3.5 Status Field

The value in the Status field shall indicate whether or not the receiver was able to execute the requested encapsulation command. A value of zero in a reply shall indicate successful execution of the command. In all requests issued by the sender, the Status field shall contain zero. If the receiver receives a request with a non-zero Status field, the request shall be ignored and no reply shall be generated.

NOTE: This field does not reflect errors that are generated by an encapsulated protocol packet contained within the data portion of the message. For example, an error encountered during an end node's processing of a Set Attributes service would be returned via the CIP specified error mechanism (see Volume 1, Chapter 3).

The status codes shall be as follows:

Table 2-3.3 Error Codes

Status Code	Description
0x0000	Success
0x0001	The sender issued an invalid or unsupported encapsulation command.
0x0002	Insufficient memory resources in the receiver to handle the command. This is not an application error. Instead, it only results if the encapsulation layer cannot obtain memory resources that it needs.
0x0003	Poorly formed or incorrect data in the data portion of the encapsulation message.
0x0004 – 0x0063	Reserved for legacy usage [1]
0x0064	An originator used an invalid session handle when sending an encapsulation message to the target.
0x0065	The target received a message of invalid length
0x0066 – 0x0068	Reserved for legacy usage [1]
0x0069	Unsupported encapsulation protocol version.
0x006A – 0xFFFF	Reserved for future expansion (Products compliant with this specification shall not use command codes in this range)

1 Error codes marked as "Reserved for legacy usage" indicate error codes that were defined prior to the publication of this specification. Their usage is undefined in this specification. Devices should not use these error codes without prior knowledge of the legacy usage.

2-3.6 Sender Context Field

The sender of the command shall assign the value in the Sender Context field of the header. The receiver shall return this value without modification in its reply. Commands with no expected reply may ignore this field.

NOTE: The sender of a command may place any value in this field. It could be used to match requests with their associated replies.

2-3.7 Options Field

The sender of an encapsulated packet shall set the options field to zero. The receiver of an encapsulated packet shall verify that the option field is zero. The receiver shall discard encapsulated packets with a non-zero option field.

NOTE: The intent of this field is to provide bits that modify the meaning of the various encapsulation commands. No particular use for this field has been specified.

2-3.8 Command Specific Data Field

NOTE: The structure of the command specific data field depends on the command code. To organize their command specific data field, most commands use either or both of the following two methods:

1) use a fixed structure
2) use the common packet format (described in section 2-6)

The common packet format allows commands to structure their command specific data field in an extensible way.

2-4 Command Descriptions

2-4.1 NOP

Either an originator or a target may send a NOP command. No reply shall be generated by this command. The data portion of the command shall be from 0 to 65511 bytes long. The receiver shall ignore any data that is contained in the message. A NOP command does not require that a session be established.

The NOP encapsulation header shall be as follows:

Table 2-4.1 NOP Header Values

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	NOP (0x00)
	Length	UINT	Length of the command specific data
	Session Handle	UDINT	Any value (ignored by target)
	Status	UDINT	0
	Sender Context	ARRAY of octet	Chosen by sender. Length of 8.
	Options	UDINT	0
Command specific data	Unused data	ARRAY of octet	Any value (ignored by target)

2-4.2 ListIdentity

2-4.2.1 General

A connection originator may use the ListIdentity command to locate and identify potential targets. This command shall be sent as a unicast message using TCP or UDP, or as a broadcast message using UDP and does not require that a session be established. The reply shall always be sent as a unicast message.

2-4.2.2 Request

The ListIdentity request shall be as shown below:

Table 2-4.2 ListIdentity Request

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	ListIdentity (0x63)
	Length	UINT	0
	Session Handle	UDINT	Any value (ignored by target).
	Status	UDINT	0
	Sender Context	ARRAY of octet	Chosen by sender. Length of 8.
	Options	UDINT	0

2-4.2.3 Reply

One reply item is defined for this command, Target Identity, with item type code 0x0C. This item shall be supported (returned) by all EtherNet/IP devices.

A receiver of the List Identity command shall reply with a standard encapsulation header and data, as shown below. The data portion of the message shall provide the information on the target's identity. The reply shall be sent to the IP address from which the broadcast request was received.

Table 2-4.3 Successful ListIdentity Reply

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	List Identity (0x63)
	Length	UINT	Length of the command specific data
	Session handle	UDINT	Any value (ignored by receiver).
	Status	UDINT	0
	Sender Context	ARRAY of octet	Value from request. Length of 8.
	Options	UDINT	0
Command specific data	Item Count	UINT	Number of target items to follow
	Target Items	STRUCT of	Interface Information
		UINT	Item ID
		UINT	Item Length
		ARRAY of octet	Item Data

The data portion of the message shall be the Common Packet Format that contains a 2-byte item count followed by an array of items providing the target identity.

The CIP Identity item defined in Table 2-4.4, shall be the first item returned. Part of this item definition follows the Get Attribute All service response definition of the Identity Object (data returned based on instance one of this object). Unlike most fields in the Common Packet Format, the Socket Address field shall be sent in big endian order.

At present no additional items are defined for the ListIdentity reply. Additional items may be defined in the future. Receivers of the ListIdentity reply shall ignore unexpected items.

Table 2-4.4 CIP Identity Item

Parameter Name	Data Type	Description
Item ID	UINT	Item ID of CIP Identity (0x0C)
Item Length	UINT	Number of bytes in item which follow (length varies depending on Product Name string)
Encapsulation Protocol Version	UINT	Encapsulation Protocol Version supported (also returned with Register Session reply).
Socket Address	STRUCT of	Socket Address (see section 2-6.3.2)
	INT	sin_family (**big-endian**)
	UINT	sin_port (**big-endian**)
	UDINT	sin_addr (**big-endian**)
	ARRAY of USINT	sin_zero (length of 8) (big-endian)
Vendor ID [1]	UINT	Device manufacturers Vendor ID
Device Type [1]	UINT	Device Type of product
Product Code [1]	UINT	Product Code assigned with respect to device type
Revision [1]	USINT[2]	Device revision
Status [1]	WORD	Current status of device
Serial Number [1]	UDINT	Serial number of device
Product Name [1]	SHORT_STRING	Human readable description of device
State [2]	USINT	Current state of device

1 These parameters are further defined by the corresponding instance attribute of the Identity Object. (see the CIP Common specification, chapter 5, Object Library)

2 The State attribute is an optional attribute of the Identity Object. If not implemented, the value shall be 0xFF

2-4.3 ListInterfaces

2-4.3.1 General

The optional List Interfaces command shall be used by a connection originator to identify non-CIP communication interfaces associated with the target. A session need not be established to send this command.

2-4.3.2 Request

The ListInterfaces request shall be as shown below.

Table 2-4.5 ListInterfaces Request

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	List Interfaces (0x64)
	Length	UINT	0
	Session Handle	UDINT	Any value (ignored by target)
	Status	UDINT	0
	Sender Context	ARRAY of octet	Chosen by sender. Length of 8.
	Options	UDINT	0

2-4.3.3 Reply

Table 2-4.6 shows the format of the successful ListInterfaces reply.

Table 2-4.6 Successful ListInterfaces Reply

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	List Interfaces (0x64)
	Length	UINT	Length of the command specific data
	Session Handle	UDINT	Any value (ignored by receiver).
	Status	UDINT	0
	Sender Context	ARRAY of octet	Value from request. Length of 8.
	Options	UDINT	0
Command specific data	Item Count	UINT	Number of target items to follow
	Target Items	STRUCT of	Interface Information
		UINT	Item ID
		UINT	Item Length
		ARRAY of octet	Item Data

The data portion of the reply contains an array of items providing interface information. The format of the data portion is a 2-byte item count followed by an array of items. At present no public items are defined for the ListInterfaces reply. If no items are included, the Item Count shall be set to 0.

Some legacy devices may return “Reserved for legacy usage items” (refer to Section 2-6). Such items shall be ignored unless the receiving device has explicit knowledge of the legacy format and usage.

2-4.4 RegisterSession

2-4.4.1 General

An originator shall send a RegisterSession command to a target to initiate a session. The RegisterSession command does not require that a session be established.

NOTE: See section 2-5, for detailed information on establishing and maintaining a session.

2-4.4.2 Request

The RegisterSession request shall be as follows:

Table 2-4.7 RegisterSession Request

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	RegisterSession (0x65)
	Length	UINT	4
	Session Handle	UDINT	Any value (ignored by target)
	Status	UDINT	0
	Sender Context	ARRAY of octet	Any value. Length of 8.
	Options	UDINT	0
Command specific data	Protocol version	UINT	1
	Options flags	UINT	No options flags are currently defined. Session options shall be set to 0 NOTE: This field is not the same as the option flags in the encapsulation header.

2-4.4.3 Reply

The target shall send a RegisterSession reply to indicate that it has registered the originator. The reply shall have the same format as the request as shown below:

Table 2-4.8 Successful RegisterSession Reply

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	RegisterSession (0x65)
	Length	UINT	4
	Session Handle	UDINT	Session Handle generated by target
	Status	UDINT	0
	Sender Context	ARRAY of octet	Value from request. Length of 8.
	Options	UDINT	0
Command specific data	Protocol version	UINT	Version from RegisterSession request if supported. If the requested version is not supported, contains the highest version supported.
	Options flags	UINT	Options flags from RegisterSession request if supported. If any requested Options flags are not supported, contains the supported Options flags.

The Session Handle field of the header shall contain a target-generated identifier that the originator shall save and insert in the Session Handle field of the header for all subsequent requests to that target. This field shall be valid only if the Status field is zero (0).

If the originator was successfully registered with the target, the Status field shall be zero (0). If the originator was not successfully registered, the Status field shall contain the appropriate error code, as follows:

- Error code 0x0001 shall be returned if the originator attempts to register more than 1 active session on the same TCP connection.
- Error code 0x0002 shall be returned if the originator does not have sufficient resources to register the originator.
- Other error codes from table 2-3.3 may be used as appropriate for general encapsulation related errors (e.g., poorly formed encapsulation message, invalid length).
- Error code 0x0069 shall be returned for Protocol Version or Options mismatches, as described below:

The Protocol Version field shall equal the requested version if the originator was successfully registered. If the target does not support the requested version of the protocol,

- the session shall not be created;
- the Status field shall be set to ‘unsupported encapsulation protocol’ (0x0069);
- the target shall return the highest supported version in the Protocol Version field.

At present, no Options flags are defined. In order to support their future definition, targets must check the value of the Options flags in the RegisterSession request. If all requested options are supported, the Options field in the reply shall contain the originator's requested value. If the target does not support the requested options,

- the session shall not be created;
- the Status field shall be set to ‘unsupported encapsulation protocol’ (0x0069);
- the target shall return the options that it supports in the RegisterSession reply.

2-4.5 UnRegisterSession

Either an originator or a target may send this command to terminate the session. The receiver shall initiate a close of the underlying TCP/IP connection when it receives this command. The session shall also be terminated when the transport connection between the originator and target is terminated. The receiver shall perform any other associated cleanup required on its end. There shall be no reply to this command, except in the event that the command is received via UDP. If the command is received via UDP, the receiver shall reply with encapsulation error code 0x01 (invalid or unsupported command).

The UnregisterSession command format shall be as follows:

Table 2-4.9 UnregisterSession Command

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	UnRegisterSession (0x66)
	Length	UINT	0
	Session Handle	UDINT	Session Handle
	Status	UDINT	0
	Sender Context	ARRAY of octet	Any value. Length of 8 (ignored by target).
	Options	UDINT	0

The receiver shall not reject the UnRegisterSession due to unexpected values in the encapsulation header (invalid Session Handle, non-zero Status, non-zero Options, or additional command data). In all cases the TCP connection shall be closed.

NOTE: See section 2-5.3 for more detail about terminating a session.

2-4.6 ListServices

2-4.6.1 General

The ListServices command shall determine which encapsulation service classes the target device supports. The ListServices command does not require that a session be established.

NOTE: Each service class has a unique type code, and an optional ASCII name.

2-4.6.2 Request

The ListServices header shall be as follows:

Table 2-4.10 ListServices Request

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	ListServices (0x04)
	Length	UINT	0
	Session Handle	UDINT	Any value (ignored by target).
	Status	UDINT	0
	Sender Context	ARRAY of octet	Chosen by sender. Length of 8.
	Options	UDINT	0

2-4.6.3 Reply

The receiver shall reply with a standard encapsulation message, consisting of the header and data, as shown below. The data portion of the message shall provide the information on the services supported.

Table 2-4.11 Successful ListServices Reply

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	ListServices (0x04)
	Length	UINT	Length of the command specific data
	Session Handle	UDINT	Any value (ignored by receiver).
	Status	UDINT	0
	Sender Context	ARRAY of octet	Value from request. Length of 8.
	Options	UDINT	0
Command specific data	Item Count	UINT	Number of items to follow
	Target Items	STRUCT of	Interface Information
		UINT	Item ID
		UINT	Item Length
		UINT	Version of encapsulated protocol shall be set to 1
		UINT	Capability flags
		ARRAY of 16 USINT	Name of service

The Item ID shall identify the service class as follows:

One service class is defined, with type code 0x100 and name "Communications". This service class shall indicate that the device supports encapsulation of CIP packets. All devices that support encapsulating CIP shall support the ListServices request and Communications service class.

NOTE: See section 2-6 for a description of items and a list of all reserved item codes.

The Version field shall indicate the version of the service supported by the target to help maintain compatibility between applications.

Each service shall have a different set of capability flags. Reserved flags shall be set to zero.

The Capability Flags, defined for the Communications service, shall be as follows:

Table 2-4.12 Capability Flags

Flag Value	Description
Bits 0 – 4	Reserved for legacy usage [1]
Bit 5	If the device supports EtherNet/IP encapsulation of CIP this bit shall be set (=1); otherwise, it shall be clear (=0)
Bits 6 – 7	Reserved for legacy usage [1]
Bit 8	Supports CIP transport class 0 or 1 UDP-based connections
Bits 9 – 15	Reserved for future expansion

1 Flags marked as “Reserved for legacy usage” indicate flags that were defined prior to the publication of this specification. Their usage is undefined in this specification. Devices should not use these flags without prior knowledge of the legacy usage. If a device receives a reserved flag that it does not understand, the reply shall be processed and the flag ignored.

The Name field shall allow up to a 16-byte, NULL-terminated ASCII string for descriptive purposes only. The 16-byte limit shall include the NULL character.

2-4.7 SendRRData

2-4.7.1 General

A SendRRData command shall transfer an encapsulated request/reply packet between the originator and target, where the originator initiates the command. The actual request/reply packets shall be encapsulated in the data portion of the message and shall be the responsibility of the target and originator.

NOTE: When used to encapsulate the CIP, the SendRRData request and response are used to send encapsulated UCMM messages (unconnected messages). See chapter 3 for more detail.

2-4.7.2 Request

The SendRRData header shall be as follows:

Table 2-4.13 SendRRData Request

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	SendRRData (0x6F)
	Length	UINT	Length of the command specific data
	Session Handle	UDINT	Session handle
	Status	UDINT	0
	Sender Context	ARRAY of octet	Chosen by sender. Length of 8.
	Options	UDINT	0
Command specific data	Interface handle	UDINT	0
	Timeout	UINT	0 to 65535
	Encapsulated packet	ARRAY of octet	see Common Packet Format specification in section 2-6)

The Interface handle shall identify the Communications Interface to which the request is directed. This handle shall be 0 for encapsulating CIP packets.

The target shall abort the requested operation after the timeout expires. When the “timeout” field is in the range 1 to 65535, the timeout shall be set to this number of seconds. When the “timeout” field is set to 0, the encapsulation protocol shall not have its own timeout. Instead, it shall rely on the timeout mechanism of the encapsulated protocol.

When the SendRRData command is used to encapsulate CIP packets, the Timeout field shall be set to 0, and shall be ignored by the target.

The encapsulated protocol packet shall be encoded in the Common Packet Format as shown in section 2-6.

2-4.7.3 Reply

The SendRRData reply, as shown below, shall contain data in response to the SendRRData request. The reply to the original encapsulated protocol request shall be contained in the data portion of the SendRRData reply.

Table 2-4.14 Successful SendRRData Reply

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	SendRRData (0x6F)
	Length	UINT	Length of the command specific data
	Session handle	UDINT	Value from request.
	Status	UDINT	0
	Sender Context	ARRAY of octet	Value from request. Length of 8.
	Options	UDINT	0
Command specific data	Interface handle	UDINT	0
	Timeout	UINT	0 to 65535 (ignored)
	Encapsulated packet	ARRAY of octet	see Common Packet Format specification in section 2-6)

The format of the data portion of the reply message shall be the same as that of the SendRRData request message.

NOTE: Since the request and reply share a common format, the reply message contains a timeout field; however, it is not used.

2-4.8 SendUnitData

The SendUnitData command shall send encapsulated connected messages. This command may be used when the encapsulated protocol has its own underlying end-to-end transport mechanism. A reply shall not be returned. The SendUnitData command may be sent by either end of the TCP connection.

NOTE: When used to encapsulate the CIP, the SendUnitData command is used to send CIP connected data in both the O⇒T and T⇒O directions.

The format of the SendUnitData command shall be as follows:

Table 2-4.15 SendUnitData Command

Structure	Field Name	Data Type	Field Value
Encapsulation header	Command	UINT	SendUnitData (0x70)
	Length	UINT	Length of data portion
	Session Handle	UDINT	Session handle
	Status	UDINT	0
	Sender Context	ARRAY of octet	Any value. Length of 8 (ignored by target).
	Options	UDINT	0
Command specific data	Interface handle	UDINT	shall be 0
	Timeout	UINT	shall be 0
	Encapsulated packet	ARRAY of octet	see Common Packet Format specification in section 2-6)

Interface handle and Timeout shall be set the zero. The timeout field is not used since no reply is generated upon receipt of a SendUnitData command.

2-5 Session Management

2-5.1 Phases of a TCP Encapsulation Session

An encapsulation session shall have three phases:

- establishing a session;
- maintaining a session;
- closing a session.

2-5.2 Establishing a Session

Session establishment shall proceed according to the following steps:

- The originator shall open a TCP/IP connection to the target, using the reserved TCP port number (0xAF12), or if specified, the TCP port number from the connection path (the means to specify an alternate TCP port number is described in chapter 3);
- The originator shall send a RegisterSession command to the target (see section 2-4.4 for a description of the RegisterSession command);
- The target shall check the protocol version in the command message to verify it supports the same protocol version as the originator. If not, the target shall return a RegisterSession reply with the appropriate Status field along with the highest supported protocol version;
- The target shall check the options flags in the command to verify that it supports the requested options. If not, the target shall return a RegisterSession reply with the appropriate Status field along with the options it supports.
- The target shall assign a new (unique) Session ID and shall send a RegisterSession reply to the originator.

Originators shall register no more than one active session on a single TCP connection.

2-5.3 Terminating a Session

Either the originator or the target may terminate the session. Sessions shall be terminated in either of two ways:

- The originator or target shall close the underlying TCP connection. The corresponding target or originator shall detect the loss of the TCP connection, and shall close its side of the connection;
- The originator or target shall send an UnRegisterSession command (see section 2-4.5 for a description of the UnregisterSession command) and shall wait to detect the closing of the TCP connection. The corresponding target or originator shall then close its side of the TCP connection. The sender of the UnRegisterSession shall detect the loss of the TCP connection, then it shall close its side of the connection.

NOTE: The second method is preferred since it results in more timely clean up of the TCP connection.

2-5.4 Maintaining a Session

Once a session is established, it shall remain established until one of the following occurs:

- the originator or target closes the TCP connection;
- the originator or target issues the UnRegisterSession command;
- the TCP connection is broken.

2-5.5 TCP Behavior (informative)

TCP is a reliable, connection-oriented protocol. If a process at either end of a connection closes its end of the connection, the TCP at the other end is notified immediately. If a message from one process to the other can not be delivered in a reasonable amount of time, the connection is assumed to be broken and an error is returned to the sender on all subsequent sends and receives on the connection.

If an originator process detects that a target has closed its end of a connection or that a connection is broken, it assumes the session with the target is broken and closes its connection to the target. A new session is then established as described above in order to resume communications with the target.

Although an originator process is notified when the other end of a connection has been closed, a broken connection can only be detected when a process actually attempts to send a message over the connection. In most cases, the originator process sends messages to targets frequently enough that a crash of a target machine is detected in a timely manner. Likewise, targets send messages back to originators frequently enough that terminated originator processes and originator machine crashes are detected quickly. However, it is possible that an originator or target may not have any messages to send on a connection for a relatively long period of time.

The TCP protocol supports keep-alive processing. An application can ask TCP to make sure the connection remains working during periods when the application does not have any messages to send. If this feature is enabled, when the connection has been idle for some period of time, TCP will send a keep-alive message to its peer at the other end of the connection. If TCP sends several keep-alive messages and does not receive a reply, TCP assumes the connection has broken and the application is notified just as if it had sent an actual message that timed out.

Most implementations of TCP/IP retry/timeout processing do not declare a failure on a connection until it has remained unusable for several minutes. This is a feature of the TCP protocol on the originator host; turning keep alives does not modify it.

2-6 Common Packet Format

2-6.1 General

The common packet format (CPF) defines a standard format for protocol packets that are transported with the encapsulation protocol. The common packet format is a general-purpose mechanism designed to accommodate future packet or address types.

The common packet format shall consist of an item count, followed by a number of items. Some items are classified as "address items" (carries addressing information) or "data items" (carries encapsulated data). The number of items to be included depends on the encapsulation command and usage of the command. Section 2-6.4 specifies the valid Common Packet Format items for the various commands and usages.

Table 2-6.1 Common Packet Format

Field Name	Data Type	Description
Item count	UINT	Number of items to follow
Item #1	Item Struct (see below)	1st CPF item
Item #2	Item Struct (see below)	2nd CPF item
…		
Item n	Item Struct (see below)	nth CPF item

The address and data item structure shall be as follows:

Table 2-6.2 CPF Item Format

Field Name	Data Type	Description
Type ID	UINT	Type of item encapsulated
Length	UINT	Length in bytes of the Data Field
Data	Variable	The data (if length >0)

Table 2-6.3 Item ID Numbers

Item ID number	Item type	Description
0x0000	address	Null (used for UCMM messages). Indicates that encapsulation routing is NOT needed. Target is either local (Ethernet) or routing info is in a data Item.
0x0001 – 0x000B		Reserved for legacy usage [1]
0x000C		ListIdentity response
0x000D – 0x0085		Reserved for legacy usage [1]
0x0086 – 0x0090		Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)
0x0091		Reserved for legacy usage [1]
0x0092 – 0x00A0		Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)
0xA1	address	Connection-based (used for connected messages)
0x00A2 – 0x00A4		Reserved for legacy usage [1]
0x00A5 – 0x00B0		Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)
0x00B1	data	Connected Transport packet
0x00B2	data	Unconnected message
0x00B3 – 0x00FF		Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)
0x0100		ListServices response
0x0101 – 0x010F		Reserved for legacy usage [1]
0x0110 – 0x7FFF		Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)
0x8000	data	Sockaddr Info, originator-to-target
0x8001	data	Sockaddr Info, target-to-originator
0x8002		Sequenced Address item
0x8003 – 0xFFFF		Reserved for future expansion of this specification (Products compliant with this specification shall not use command codes in this range)

1 Items marked as "Reserved for legacy usage" indicate Item IDs that were defined prior to the publication of this specification. Their behavior is undefined in this specification. Devices should not use these Item IDs without prior knowledge of the legacy usage.

2-6.2 Address Items

2-6.2.1 Null Address Item

The null address item shall contain only the type id and the length as shown below. The length shall be zero. No data shall follow the length. Since the null address item contains no routing information, it shall be used when the protocol packet itself contains any necessary routing information. The null address item shall be used for Unconnected Messages.

Table 2-6.4 Null Address Item

Field Name	Data Type	Field Value
Type ID	UINT	0
Length	UINT	0

2-6.2.2 Connected Address Item

This address item shall be used when the encapsulated protocol is connection-oriented. The data shall contain a connection identifier.

NOTE: Connection identifiers are exchanged in the Forward_Open service of the Connection Manager.

Table 2-6.5 Connected Address Item

Field Name	Data Type	Field Value
Type ID	UINT	0xA1
Length	UINT	4
Data	UDINT	Connection Identifier

2-6.2.3 Sequenced Address Item

This address item shall be used for CIP transport class 0 and class 1 connected data. The data shall contain a connection identifier and a sequence number.

Table 2-6.6 Sequenced Address Item

Field Name	Data Type	Field Value
Type ID	UINT	0x8002
Length	UINT	8
Data	UDINT	Connection Identifier
	UDINT	Sequence Number

2-6.3 Data Items

2-6.3.1 Unconnected Data Item

The data item that encapsulates an unconnected message shall be as follows:

Table 2-6.7 Unconnected Data Item

Field Name	Data Type	Field Value
Type ID	UINT	0xB2
Length	UINT	Length, in bytes, of the unconnected message
Data	Variable	The unconnected message

NOTE: The format of the “data” field is dependent on the encapsulated protocol. When used to encapsulate CIP, the format of the “data” field is that of a Message Router request or Message Router reply. See chapter 3 of this specification for details of the encapsulation of UCMM messages. See Volume 1, Chapter 2 for the format of the Message Router request and reply packets.

The context field in the encapsulation header shall be used for unconnected request/reply matching.

2-6.3.2 Connected Data Item

The data item that encapsulates a connected transport packet shall be as follows:

Table 2-6.8 Connected Data Item

Field Name	Data Type	Field Value
Type ID	UINT	0xB1
Length	UINT	Length, in bytes, of the transport packet
Data	Variable	The transport packet

NOTE: The format of the "data" field is dependent on the encapsulated protocol. When used to encapsulate CIP, the format of the "data" field is that of connected packet. See chapter 3 of this specification for details of the encapsulation of connected packets. See chapter 3 of the CIP Specification (Volume 1) for the format of connected packets.

2-6.3.3 Sockaddr Info Item

The Sockaddr Info items shall be used to communicate IP address or port information necessary to create Class 0 or Class 1 connections. There are separate items for originator-to-target and target-to-originator socket information. The items are present as additional data in Forward_Open / Large_Forward_Open request and reply services encapsulated in a SendRRData message. Volume 2, Chapter 3-3.9 describes the usage of these items in the context of creating CIP connections.

The Sockaddr Info items shall have the following structure:

Table 2-6.9 Sockaddr Item

Field Name	Data Type	Field Value
Type ID	UINT	0x8000 for O⇒T, 0x8001 for T⇒O
Length	UINT	16 (bytes)
sin_family	INT	shall be AF_INET = 2. This field shall be sent in big endian order.
sin_port	UINT	For point-point connections, sin_port shall be set to the UDP port to which packets for this CIP connection will be sent. For point-point connections, it is recommended that the registered UDP port (0x8AE) be used. When used with a multicast connection, the sin_port field shall be set to the registered UDP port number (0x08AE) and treated by the receiver as "don't care". This field shall be sent in big endian order.
sin_addr	UDINT	For multicast connections, sin_addr shall be set to the IP multicast address to which packets for this CIP connection will be sent. When used with a point-point connection, the sin_addr field shall be treated by the receiver as "don't care". It is recommended that the sender set sin_addr to 0 for point-point connections. This field shall be sent in big endian order.
sin_zero	ARRAY of USINT	Length of 8. Recommended value of zero; not enforced

NOTE: The structure of the Sockaddr item has been patterned after the sockaddr_in structure from the Winsock specification, version 1.1.

2-6.4 Valid Common Packet Format Item Usage Summary

The Common Packet Format is used with the following encapsulation commands:

- ListIdentity reply
- ListInterfaces reply
- ListServices reply
- SendRRData request and reply
- SendUnitData
- Transport class 0 and class 1 packets (no encapsulation header)

Table 2-6.10 shows the valid usage of CPF items for the various encapsulation commands. Note that Chapter 3 of this volume contains the detailed formats and usage for CIP connected and unconnected messages.

Table 2-6.10 Usage of CPF items

Command	Required CPF Items	Optional CPF Items	Action if unexpected or undefined CPF items are present
ListIdentity reply	ListIdentity reply item	None.	Ignore items.
ListInterfaces reply	None.	Legacy devices may return 'Reserved for legacy usage' items.	Ignore items.
ListServices reply	ListServices item for the "Communications" service.	Legacy devices may return 'Reserved for legacy usage' items.	Ignore items.
SendRRData request	Address item followed by Data item.	When used to encapsulate the Forward_Open service, additional Sockaddr_info items may be present, as specified in Chapter 3.	Return error (0x0003)
SendRRData reply	Address item followed by Data item.	When used to encapsulate the Forward_Open reply, additional Sockaddr_info items may be present, as specified in Chapter 3.	Signal error to the calling application. For Forward_Open reply, connection shall not be established at the originator.
SendUnitData	Address item followed by Data item	None.	Discard message.
Class 0/1 packet	Address item followed by Data item	None.	Discard message.

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 3: Mapping of Explicit and I/O Messaging to TCP/IP

Contents

3-1 Introduction

This chapter (chapter 3) of the EtherNet/IP specification describes the application of the encapsulation in chapter 2 to the Common Industrial Protocol (CIP). Specifically, this chapter documents the encapsulation of the UCMM and connected packets; extends the format of the path to include IP addresses; and limits which CIP transport parameters can be used in combination.

3-2 CIP Packets over TCP/IP

When the path of a CIP packet traverses an Ethernet-TCP/IP network, the encapsulated packet shall be transmitted using the TCP/IP protocol suite and the encapsulation protocol defined in chapter 2.

3-2.1 Unconnected Messages

UCMM packets shall be transmitted over a TCP/IP connection, using the encapsulation protocol defined in chapter 2. A UCMM request shall be formatted as shown in Table 3-2.1.

Table 3-2.1 UCMM Request

Structure	Field Name		Data Type	Field Value
Encapsulation header	Command		UINT	SendRRData (0x6F)
	Length		UINT	Length of command specific data portion
	Session handle		UDINT	Handle returned by RegisterSession
	Status		UDINT	0
	Sender Context		ARRAY of 8 octet	Chosen by sender
	Options		UDINT	0
Command specific data	Interface handle		UDINT	0
	Timeout		UINT	Any value (ignored by target)
	Encapsulated packet (in the Common Packet Format)	Item count	UINT	2 (indicates 1 address item, 1 data item)
		Address Type ID	UINT	0 (Null Address Item)
		Address Length	UINT	0
		Data Type ID	UINT	0x00B2 (Unconnected Data Item)
		Data Length	UINT	Length of the next field in bytes (length of the MR request packet)
		MR request packet	ARRAY of USINT	This field contains a CIP Message Router request packet as defined in Volume 1, Chapter 2.

Likewise, the UCMM reply shall be formatted as shown in Table 3-2.2.

Table 3-2.2 UCMM Reply

Structure	Field Name		Data Type	Field Value
Encapsulation header	Command		UINT	SendRRData (0x6F)
	Length		UINT	Length of command specific data portion
	Session handle		UDINT	Handle returned by RegisterSession
	Status		UDINT	0
	Sender Context		ARRAY of 8 octet	copied from the corresponding UCMM request
	Options		UDINT	0
Command specific data	Interface handle		UDINT	0
	Timeout		UINT	Any value (ignored by receiver)
	Encapsulated packet (in the Common Packet Format)	Item count	UINT	2 (indicates 1 address item, 1 data item)
		Address Type ID	UINT	0 (Null Address Item)
		Address Length	UINT	0
		Data Type ID	UINT	0x00B2 (Unconnected Data Item)
		Data Length	UINT	Length of the next field in bytes (length of the MR response packet)
		MR response packet	ARRAY of USINT	This field contains a CIP Message Router reply packet as defined in Volume 1, Chapter 2.

Note: Chapter 2 (Encapsulation Protocol) defines the valid format for the SendRRData and other encapsulation commands.

The maximum size of the MR request and MR response packet items in the SendRRData command is 504 bytes (the maximum size on ControlNet), to ensure that a UCMM message can traverse all the links in a CIP network path. Devices may support the Large_Forward_Open service (see Volume 1, Chapter 3) to allow more efficient access to large application data sizes.

When a target receives a SendRRData greater than the maximum size, it shall return an error response with encapsulation error code 0x65 (target received a message of invalid length). Error code 0x03 is allowed for existing implementations but shall be considered to be "deprecated". New implementations shall use 0x65.

3-2.2 CIP Transport Class 0 and Class 1 Connections

NOTE: Please see Volume 1 for the definition and usage of CIP transport class 0 and class 1 connections.

3-2.2.1 CIP transport Class 0 and Class 1 Packets

Packets for CIP transport class 0 and class 1 connections shall be transmitted using UDP. Packets for multicast connections shall be transmitted using IP multicast. The packet shall be formatted as shown in Table 3-2.3. Note that the packets use the Common Packet Format defined in Chapter 2, but without the encapsulation header.

Table 3-2.3 UDP Data Format for Class 0 and Class 1

Field Name	Type	Value
Item Count	UINT	2
Type ID	UINT	0x8002 (Sequenced Address Type)
Length	UINT	8
Address Data	UDINT	Connection ID (from Forward_Open reply)
	UDINT	Encapsulation Sequence Number. Note: this is different from the sequence count in the transport class 1 packet. Refer to section 3-4.1.
Type ID	UINT	0x00B1 (Connected Data Type)
Length	UINT	Number of bytes in packet to follow
Data		Transport class 0 or class 1 packet as defined in Volume 1.

3-2.2.2 Behavior of Class 0 and Class 1 Connections (informative)

Since Ethernet does not have a mechanism for sending scheduled data, several important aspects of class 0 and class 1 behavior are noted as follows:

On Ethernet, it is possible for a CIP transport class 0 or class 1 connected data packet to be lost, for example due to excessive collisions. By definition, class 0 and class 1 connections do not guarantee the delivery of every packet. Rather, producers simply send data at the specified rate (the API). If a packet is lost on a class 0 or class 1 connection, the consumer receives the next packet from the producer.

The degree to which lost packets can be tolerated is application-specific. Ethernet is not suitable for those applications that cannot tolerate any lost packets.

For CIP transport class 1 connections, the consuming CIP transport can detect packet loss by examining the CIP sequence number in the class 1 packet. For class 0 connections, it is not possible for the application to know that a specific packet has been lost since class 0 does not use a sequence number.

The connection timeout mechanism provides feedback to the application when too many packets are lost. The connection timeout is determined by the Requested Packet Interval (RPI) and by the Connection Timeout Multiplier. If a packet is not received in the time specified by the RPI times the Connection Timeout Multiplier, the connection is broken. For example, if the RPI is 50 ms and the Connection Timeout Multiplier is 4, then the connection will time out if a fresh packet is not received in 200 ms (the equivalent of 4 packets being lost). Receipt of older packets (those with lower CIP sequence numbers) will not sustain the CIP connection.

The degree of packet loss for any particular connection will be dependent upon many factors related to the user's Ethernet network configuration. It is beyond the scope of this specification to address this in further detail.

3-2.2.3 No Linkage with TCP Connections

In order to open a CIP transport class 0 or 1 connection, a TCP connection and an EtherNet/IP encapsulation session must first be established. The TCP connection is used to send the Forward_Open service and receive the Forward_Open response. Once the TCP connection is opened, and CIP transport class 0 and 1 connections are established, it is recommended that EtherNet/IP devices leave the TCP connection open. If the TCP connection is left open, it is then available for subsequent communications such as a Forward_Close or other explicit messages.

Although it is recommended that devices leave the TCP connection open, there shall be no linkage between the TCP connection used to open a transport class 0 or class 1 connection and the resulting class 0 or class 1 connection. If a TCP connection closes, the closing of the TCP connection shall not cause the target or the originator to close any corresponding CIP transport class 0 or class 1 connections.

3-2.3 CIP Transport Class 2 and Class 3 Connections

CIP transport class 2 and class 3 connected data shall be sent over a TCP connection, using the SendUnitData encapsulation command. The connected data shall be formatted as shown in Table 3-2.4.

Table 3-2.4 Transport Class 2 and Class 3 Connected Data

Structure	Field Name		Data Type	Field Value
Encapsulation header	Command		UINT	SendUnitData (0x70)
	Length		UINT	Length of command specific data portion
	Session handle		UDINT	Handle returned by RegisterSession
	Status		UDINT	0
	Sender Context		ARRAY of 8 octet	Any value (ignored by target)
	Options		UDINT	0
Command specific data	Interface handle		UDINT	0
	Timeout		UINT	Any value (ignored by target)
	Encapsulated packet (in the Common Packet Format)	Item count	UINT	2 (indicates 1 address item, 1 data item)
		Address Type ID	UINT	0xA1 (Connected Address item)
		Address Length	UINT	4
		Address Data	UDINT	Connection ID from Forward_Open/Forward_Open_Reply
		Data Type ID	UINT	0x00B1 (Connected Data item)
		Data Length	UINT	Length of the next field in bytes (i.e., length of the transport class 2/3 PDU, including the sequence count).
		Data	ARRAY of USINT	The transport class 2/3 PDU (including the sequence count) as defined in Volume 1.

Multiple CIP connections may be sent over a single TCP connection. An implementation need not support a specific number of CIP connections per TCP connection. An implementation may impose an upper bound if it chooses.

Because of the full-duplex nature of TCP, the CIP originator to target (O⇒T) and CIP target to originator (T⇒O) link connections shall use the same TCP connection. However if a target subsequently originates a CIP connection, then it shall be considered an originator, and a different TCP connection shall be used.

NOTE: This standard defines no requirements for management of the TCP connection, such as inactivity timeouts, or closing the TCP connection when all native connections are closed. However, implementations are free to implement these.

Targets and originators shall close any CIP transport class 2 or 3 connections when the corresponding originating TCP connection is closed.

3-2.4 CIP Transport Classes 4 Through 6

The encapsulation protocol described in chapter 2 shall not be used to encapsulate CIP transport classes 4, 5 and 6.

3-3 Connection Manager Object

3-3.1 Connection Parameters

NOTE: This section documents the Connection Manager parameters that have requirements specific to the TCP/IP encapsulation. Connection Manager parameters are fully described in Volume 1, Chapter 3.

3-3.2 Connection Type

The CIP connection type shall be NULL, MULTICAST, or POINT2POINT. The MULTICAST connection type shall be supported only for CIP transport class 0 and class 1 connections.

3-3.3 Priority

Volume 1, Chapter 3 defines 4 levels of CIP priority: LOW, HIGH, SCHEDULED, URGENT, with URGENT being the highest priority. Section 3-7 defines Quality of Service (QoS) behavior with respect to the different CIP priority levels.

3-3.4 Trigger Type

The CIP trigger type shall be CYCLIC, CHANGE_OF_STATE, or APPLICATION. CIP transport class 0 and class 1 connections that use CHANGE_OF_STATE triggering shall use the Production Inhibit Time segment (see Volume 1).

3-3.5 Connection Size

The CIP connection size shall be no larger than 65511 bytes.

NOTE: The Forward_Open request limits the connection size to 511 bytes; however, the optional Ex_Forward_Open allows larger connection sizes.

3-3.6 Connection Request Timeout

To reliably establish a CIP connection that extends onto a TCP/IP link, the connection request time-out shall be large enough to allow the connection to be established, which could involve resolving a host name, or going through multiple gateways.

Because of the large variation in connection request processing over TCP/IP, CIP routers in the connection path shall not subtract anything from the connection request timeout.

3-3.7 Connection Path

The link address portion of a TCP/IP connection path segment shall be encoded within a port segment as a string of ASCII characters. The following forms shall all be supported:

- IP address in dot notation, for example “130.151.132.55” (see RFC 1117 for the format of IP addresses);
- IP address in dot notation, followed by a ":" separator, followed by the TCP port number to be used at the specified IP address;
- Host name, for example “plc.controlnet.org”. The host name shall be resolved via a DNS request to a name server (see RFC 1035 for information on host names and name resolution);
- Host name, followed by a ":" separator, followed by the TCP port number to be used at the specified host.

The port number shall be represented in either hex or decimal. Hex shall be indicated by a leading "0x". When a port number is specified, it shall be used rather than the standard port number used for the encapsulation protocol (0xAF12). Only port 0xAF12 is guaranteed to be available in an EtherNet/IP compliant device.

NOTE: Other TCP port numbers may be implemented; however, this specification does not provide a mechanism to determine which TCP port numbers are supported by a device. The use of other TCP port numbers is therefore discouraged. The guaranteed TCP port number, 0xAF12, has been reserved with the Internet Assigned Numbers Authority (IANA) for use by the encapsulation protocol.

Since port segments must be word-aligned, a pad byte may be required at the end of the string. The pad byte shall be 0x00, and shall not be counted in the Optional Address Size field of the port segment.

NOTE: Examples of port segments are shown in Table 3-3.1 (see Volume 1 for the definition of a port segment).

Table 3-3.1 TCP/IP Link Address Examples

Port Segment	IP address	Notes
[12][0D] [31][33][30][2E] [31][35][31][2E] [31][33][32][2E][31][00]	130.151.132.1	Multi-byte address for port 2, 13 byte string plus a pad byte
[13][12] [70][6C][63][2E] [63][6F][6E][74][72][6F][6C][6E][65][74] [2E] [6F][72][67]	plc.controlnet.org	Multi-byte address for port 3, 18 byte string, no pad byte
[16][15] [31][33][30][2E] [31][35][31][2E] [31][33][32][2E] [35][35][3A] [30][78][33][32][31][30][00]	130.151.132.55:0x3210	Multi-byte address for port 6, 21 byte string plus a pad byte
[15][17] [70][6C][63][2E] [63][6F][6E][74][72][6F][6C][6E][65][74] [2E] [6F][72][67][3A] [39][38][37][36][00]	plc.controlnet.org:9876	Multi-byte address port 5, 23 byte string plus a pad byte

3-3.7.1 Network Connection ID

3-3.7.1.1 General

For EtherNet/IP connections, the Network Connection ID shall be a 32-bit identifier meaningful to the device that chooses it. The Network Connection ID need not be subdivided into any specific fields.

In general, the consuming device selects the Network Connection ID for a point-to-point connection, and the producing device selects the Network Connection ID for a multicast connection. The following table shows which device, Target or Originator, shall choose the T->O and O->T Network Connection IDs:

Table 3-3.2 Network Connection ID Selection

Connection Type	Which Network Connection ID	Who chooses Connection ID
Point-to-point	Originator -> Target	Target
	Target -> Originator	Originator
Multicast	Originator -> Target	Originator
	Target -> Originator	Target

The Network Connection ID shall not be reused until the connection has been closed or has timed out. When a device restarts, it shall not reuse Network Connection IDs from previously opened connections until those connections have been closed or have timed out. A specific connection ID shall not be reused so long as there is the possibility that packets with that connection ID are present in the network.

The following two sections describe possible methods to implement unique Network Connection IDs.

3-3.7.1.2 Using an Incarnation ID (informative)

This section describes one solution that prevents Connection ID reuse when a device restarts. With this solution, the Ethernet device generates Connection IDs for class 0 and class 1 connections with format shown in Figure 3-3.1.

Figure 3-3.1 Connection ID with Incarnation ID

Where:

Connection Number is a 16-bit identifier meaningful to the device choosing the Connection ID.

Incarnation ID is a 16-bit identifier that each device generates before accepting or initiating any connections.

The Incarnation ID persists while the device is powered up and accepting connections. Each successive power-up cycle must cause a new (unique) Incarnation ID to be generated. The following are acceptable methods for generating Incarnation ID's:

Devices may generate a unique Incarnation ID by saving the Incarnation ID in non-volatile storage: when the device powers up, it reads the Incarnation ID from non-volatile storage. This is the Incarnation ID to use for the current cycle. It then increments the Incarnation ID and stores it for the next cycle. Note, however, that non-volatile memory devices generally have a limit to the number of times the device may be written. Depending on the device, it may not be feasible to write the Incarnation ID each powerup.

Devices may generate a unique Incarnation ID by generating a pseudo-random number at powerup. This approach requires care. By definition, there is a non-zero probability that the generated Incarnation ID is the same as the previous one. However, if done wisely, the probability is small enough to not be a concern.

Devices such as workstations, because of the large variation in startup time, can safely use the value of the system clock as an Incarnation ID. However, for embedded devices, using the system clock is not reliable since the firmware generally goes through the exact same sequence of instructions at each powerup. This results in the same clock value at the point where it would be selected for the Incarnation ID. For these embedded devices, the Incarnation ID needs to be generated based on random inputs. This is best done using a pseudo-random number generator such as the MD5 algorithm.

3-3.7.1.3 Pseudo-Random Connection ID Per Connection (informative)

This section describes another solution that prevents Connection ID reuse when a device restarts. With this solution, the device generates a pseudo-random Connection ID each time a class 0 or class 1 Connection ID is needed. The Connection ID format for this approach is shown in Figure 3-3.2.

Figure 3-3.2 Pseudo-Random Connection ID

Where:

Connection Number is a 16-bit identifier meaningful to the device choosing the Connection ID.

Pseudo-Random Number is a 16-bit number generated using an appropriate pseudo-random number generator.

With this approach, the device generates the Pseudo-Random Number portion each time a Connection ID is needed. A "strong mixing function" such as the MD5 algorithm [RFC 1321] [RFC 1750] should be used to generate the pseudo-random number. Such functions take multiple input and produce pseudo-random outputs.

In order to prevent Connection IDs from being reused across powerups, the seed values for the inputs to the MD5 algorithm must be unique across successive powerup cycles. The recommended approach is to use the following inputs upon receiving the first incoming connection request:

- Vendor ID, Serial Number, Connection Serial Number
- Contents of the sockaddr_in struct (for the next hop if outbound connection; of the sender if inbound connection)
- Value of system clock

NOTE: This assumes the Ethernet device is a bridge or the Target of the connection and would not be applicable if the device is the connection Originator. For connection originators, the above seed values would likely be the same across successive powerups. Connection originators must use another source for initial seed values, or else use the Incarnation ID approach.

By definition, there is a non-zero probability that a Connection ID conflict may still occur. However, the probability is lowered by:

Using a robust pseudo-random number generator such as the MD5 algorithm.

Ensuring the seed values are different on successive power-up cycles.

3-3.8 Forward_Open for CIP Transport Class 2 and Class 3 Connections

The Forward_Open service for CIP class 2 and class 3 connections shall be sent over a TCP connection using the SendRRData command defined in chapter 2. Sockaddr Info items included within a Forward_open or Forward_open_reply packet that establishes a class 2 or class 3 connection shall be ignored. When Sockaddr Info items are included, it is recommended that the sender set the sin_port and sin_addr fields set to 0.

3-3.9 Forward_Open for CIP Transport Class 0 and Class 1 Connections

3-3.9.1 Use of TCP for Class 0 and Class 1 Forward_Open

The Forward_Open service for CIP transport class 0 and class 1 connections shall be sent over a TCP connection using the SendRRData command defined in chapter 2.

3-3.9.2 General Use of Sockaddr Info Items

As part of the Forward_Open dialog, the producer and consumer shall exchange the UDP port numbers and IP multicast address (for multicast connections) necessary to send the CIP transport class 0 and class 1 connected data. The Sockaddr Info item defined in Chapter 2 shall be used to encode the UDP port numbers and IP multicast address. The inclusion and use of a Sockaddr Info item varies depending on whether the connection is multicast or point-to-point, and whether the connection originator or the connection target is the multicast producer.

3-3.9.3 Sockaddr Info Item Placement and Errors

The Sockaddr Info item(s) shall be placed after the Forward_Open and/or Forward_Open_reply data in the SendRRData command/reply. It shall be considered an error if a Sockaddr Info item is not present for a multicast connection, does not have the correct sin_family value of AF_INET = 2, or specifies field values which are illegal for the intended usage.

The receiver of a Forward_open_request shall return a Forward_open_reply with a status code 0x01 and extended status 0x205 (Parameter error in unconnected service) for Sockaddr Info items containing errors specified in the preceding paragraph. The receiver of a Forward_open_reply containing any Sockaddr Info items with errors shall consider the entire reply to be in error.

3-3.9.4 Use of Sockaddr Info Item for Multicast Connections

For multicast connections, the multicast producer shall choose an IP multicast address to which to send the connected data. The port number shall be the registered UDP port number (0x08AE) assigned by the IANA. The multicast consumer shall receive the connected data on the registered port number. A Sockaddr Info item shall be sent with the Forward_open (if the O->T Connection Type is multicast), or with the Forward_open_reply (if the T->O Connection Type is multicast). The chosen IP multicast address shall be encoded via the sin_addr field of the Sockaddr Info item. The sin_port field of the Sockaddr Info item shall be set to 0x08AE and treated by the receiver as “don’t care”.

3-3.9.5 Use of Sockaddr Info Item for Point-Point Connections

For point-point connections, the point-point consumer shall choose a UDP port number on which it will receive the connected data. The point-point producer shall send the connected data to the port number chosen by the point-point consumer. It is recommended that the consumer use the registered port number (0x08AE), however the consumer may alternatively choose a different port number.

Note: using a port number other than 0x08AE has potential implication for Quality of Service (QoS) configuration and management in infrastructure devices. Switches that have been configured to prioritize packets with port number 0x8AE may not prioritize EtherNet/IP packets with a different port number.

If the point-point consumer chooses a port number that is different than 0x08AE, then the point-point consumer shall send a Sockaddr Info item indicating the chosen port number. The Sockaddr Info item shall be sent with the Forward_open (if the T->O Connection Type is point-point), or with the Forward_open_reply (if the O->T Connection Type is point-point). The sin_port field of the Sockaddr Info item shall contain the port number selected by the consumer. The sin_addr field of the Sockaddr Info item shall be treated as a “don’t care” by the receiver of the Forward_open or Forward_open_reply. It is recommended that the sender set the sin_addr field to zero.

If the point-point consumer elects to use the registered port 0x08AE to receive connected data, the consumer is not required to send a Sockaddr Info item. The consumer may optionally include a Sockaddr Info item as described in the preceding paragraph, with sin_port field containing the registered port number.

For a point-point connection the sin_addr field of the Sockaddr Info item shall be treated as a “don’t care” by the receiver of the Forward_open or Forward_open_reply. It is recommended that the sender set the sin_addr field to zero.

3-3.9.6 Usage Summary of Sockaddr Info for Class 0 or Class 1 Connections

Table 3-3.3 shows the usage of Sockaddr Info items in transport class 0 and 1 Forward_Open and Forward_Open response. In the cases where the item is “ignored if present”, devices shall not check the contents of the Sockaddr Info item for validity.

Table 3-3.3 Sockaddr Info Usage

Connection Type	Service	Sockaddr Info items
Point-Point O->T Multicast T->O	Forward_Open	O->T ignored if present T->O ignored if present
	Forward_Open response	O->T item optional (registered port number used if item is not present) T->O item required
Multicast O->T Point-Point T->O	Forward_Open	O->T item required T->O item optional (registered port number used if item is not present)
	Forward_Open response	O->T ignored if present T->O ignored if present
Point-Point O->T Point-Point T->O	Forward_Open	O->T ignored if present T->O item optional (registered port number used if item is not present)
	Forward_Open response	O->T item optional (registered port number used if item is not present) T->O ignored if present
Multicast O->T Multicast T->O	Forward_Open	O->T item required T->O ignored if present
	Forward_Open response	O->T ignored if present T->O item required

3-3.9.7 Mapping Connections to IP Multicast Addresses

NOTE: It is recommended, though not required, that producers use a unique IP multicast address for each active multicast connection. Depending upon the implementation, this can reduce the amount of connection screening on the part of the consumer. It also allows the consumer to more evenly service incoming connected data from multiple connections.

Since a unique IP multicast address per multicast connection is not required, consumers shall be able to handle the situation in which packets from multiple multicast connections are being sent to the same IP multicast address. Consumers shall be able to screen the incoming packets based on the Connection ID and source IP address.

NOTE: Requirements for screening connected data are defined in section 3-4.2.

3-3.9.8 Completing the Multicast Connection (informative)

After receiving the Forward_Open_reply the consuming Ethernet devices should join the desired IP Multicast Group in order to receive the IP Multicast datagrams. The exact method for doing this depends on the TCP/IP application programming interface in use on the device.

3-4 CIP Transport Class 0 and Class 1 Connected Data

3-4.1 CIP Transport Class 0 and Class 1 Packet Ordering

NOTE: By definition, CIP class 0 and class 1 transports do not detect out-of-order packets. For class 0, every packet is considered to be new data. For class 1, only duplicate data is detected. A received packet is considered to be new data whenever the sequence count is different (either greater or lesser) than the previous packet's sequence count.

NOTE: When using UDP to transport CIP class 0 and class 1 connected data, there is no guarantee that packets arrive in the same order that they were sent. When both sender and receiver are on the same subnet, packets typically arrive in order. However, when going through routers, when there are multiple paths that a packet could take, it is possible for packets to arrive out of order.

For class 0 and class 1 connections over EtherNet/IP, devices shall maintain an encapsulation sequence number in the UDP payload defined in section 3-2.2.1. The encapsulation sequence number shall be maintained per connection. Each time an EtherNet/IP device sends a CIP class 1 packet, it shall increment the encapsulation sequence number for that connection. If the receiving EtherNet/IP device receives a packet whose encapsulation sequence number is less than the previously received packet, the packet with the smaller encapsulation sequence number shall be discarded.

The sequence number shall be operated on with modular arithmetic to deal with sequence rollover.

NOTE: Dealing with 32-bit sequence numbers is described in RFC793 (the TCP definition), as follows:

It is essential to remember that the actual sequence number space is finite, though very large. This space ranges from 0 to 2**32 - 1. Since the space is finite, all arithmetic dealing with sequence numbers must be performed modulo 2**32. This unsigned arithmetic preserves the relationship of sequence numbers as they cycle from 2**32 - 1 to 0 again. There are some subtleties to computer modulo arithmetic, so great care should be taken in programming the comparison of such values. The symbol "=<" means "less than or equal" (modulo 2**32).

Example macros show how this may be done:

```
/*
 * TCP sequence numbers are unsigned 32 bit integers operated
 * on with modular arithmetic.  These macros can be
 * used to compare such integers.
 */

#define SEQ_LT(a,b)     ((int)((a)-(b)) < 0)
#define SEQ_LEQ(a,b)    ((int)((a)-(b)) <= 0)
#define SEQ_GT(a,b)     ((int)((a)-(b)) > 0)
#define SEQ_GEQ(a,b)    ((int)((a)-(b)) >= 0)
```

3-4.2 Screening Incoming Connected Data

Ethernet devices that receive class 0 and class 1 connected data shall screen incoming packets based on the network connection ID and IP address of the sending device. This is necessary for the following reasons:

- For multicast connections, there is no guaranteed mechanism to prevent multiple devices from using the same IP multicast address. Consequently, a device could receive (bogus) multicast connected data from a device with which it has not established a connection.
- For multicast connections, a device is allowed to use the same IP multicast address for multiple class 0 and class 1 multicast connections.
- To prevent network connection ID conflicts.

When a class 0 or class 1 connection is established, the target and originating Ethernet devices shall record the network connection ID on which they will receive connected data, coupled with the IP address of the device at the other end of the connection. When a device receives connected data, it shall confirm that the network connection ID is valid for the IP address of the sending device. If not, the packet shall be discarded.

3-5 IP Multicast Scoping and Address Allocation

3-5.1 Background (informative)

3-5.1.1 General

Two issues related to IP multicast must be considered when implementing EtherNet/IP multicast connections: IP multicast scoping and IP multicast address allocation.

IP multicast scoping refers to the practice of limiting how widely a given multicast datagram is propagated across the network. IP multicast address allocation refers to the problem of how applications select IP multicast addresses that are used to send and receive IP multicast datagrams.

The following subsections on multicast scoping and allocation practices are informative, and are intended to set the general context for considering the issues of scoping and address allocation. Specific requirements for EtherNet/IP devices follow in subsequent sections.

3-5.1.2 IP Multicast Scoping Practices

In general, most currently deployed networks use the practice of “TTL scoping” in conjunction with router and/or switch configuration to confine multicast traffic to desired network boundaries.

TTL scoping refers to the practice of using the “Time to Live” (TTL) field in the IP header to limit the number of network hops over which the multicast packet is propagated. When sending an IP multicast datagram, a host can set the TTL field in the IP header to an appropriate value based on how widely the datagram should be propagated. As the datagram is routed through the network, each hop decrements the TTL field. Routers can be configured with TTL thresholds such that they will not forward a packet unless the TTL is greater than the threshold.

Note that a multicast datagram with an initial TTL of 1 limits the datagram to the local subnet. Other common TTL values are 16 for multicast within a site and 64 for multicast within a region.

In addition to TTL scoping, multicast routing protocols and other methods are commonly used to control the propagation of multicast traffic. Routers commonly support multicast protocols such as PIM, DVMRP, etc. Switches that implement "IGMP snooping" can limit the multicast packets sent on a port to only those multicast addresses for which the end device has issued an IGMP membership message. Configuration of switches and routers is usually done by knowledgeable staff.

3-5.1.3 IP Multicast Address Allocation Practices

The entire IP multicast address space is 224.0.0.0 through 239.255.255.255. The Internet Assigned Numbers Authority (www.iana.org) is responsible for allocation of the IP multicast address space. IP multicast addresses have been assigned to particular organizations, and for particular protocols. In addition there is a large block of IP multicast addresses allocated for "administratively scoped" multicast, from which applications may allocate addresses, and for which a suite of allocation and scoping protocols are being developed by the Internet Engineering Task Force (IETF). The administratively scoped range is from 239.0.0.0 through 239.255.255.255 (and is further partitioned into additional ranges).

Unfortunately at present there are no widely deployed standard mechanisms for allocating and assigning multicast addresses to applications. For example, when a network administrator deploys a video streaming application, the application will have its own specific mechanism for assigning IP multicast addresses.

3-5.2 Multicast Scoping for EtherNet/IP

By default, EtherNet/IP devices shall use a TTL equal to 1 for transport class 0 and 1 multicast packets. The use of a TTL value of 1 prevents multicast packets from propagating beyond the local subnet. When TTL is equal to 1, both the EtherNet/IP producer and consumer must be on the same subnet.

EtherNet/IP devices are strongly encouraged to support the explicit configuration of the TTL value for IP multicast packets. If supported, devices shall use the TCP/IP Interface Object (class 0xF5) as the mechanism to configure the TTL value. When a TTL value greater than 1 is configured, then the producer and consumer may be on different subnets.

If the TTL value has not been configured to be greater than 1 and if a multicast connection request is received from an originator on a different subnet, then the device shall return General Status 0x01 and Extended Status 0x813 in the Forward_Open Reply.

When the TTL value is explicitly configured, it shall be used for all EtherNet/IP multicast packets.

3-5.3 Multicast Address Allocation for EtherNet/IP

EtherNet/IP defines two mechanisms for allocation of IP multicast addresses used for EtherNet/IP multicast packets: using an algorithm based on the device's IP address, and explicit configuration via the TCP/IP Interface Object. EtherNet/IP devices shall implement the algorithm-based method by default. Devices are also strongly encouraged to implement the method for explicit configuration of multicast addresses. Both methods are described below:

1. Allocation algorithm based on the device's IP address:

   The overall IP multicast address range shall be the Organizational Local Scope, and shall start at 239.192.1.0. Each device shall use a block of (at most) 32 multicast addresses from this range. Each device shall calculate the block of multicast addresses via an algorithm, described further below, that uses the Host Id portion of the device's IP address.

   A device's Host Id shall be determined by applying the subnet mask to the device's IP address. If a subnet mask is configured for the device, the subnet mask shall be applied to the IP address to determine the Host Id. If no subnet mask is in use, then the class of the device's IP address shall be used to determine the Host Id (e.g., for Class C addresses 8 bits of Host Id shall be used, for Class B addresses 16 bits of host id shall be used, and for Class A addresses 24 bits of Host Id shall be used).

   In order to keep the IP multicast addresses within the IPv4 Organization Local Scope, and to put a reasonable bounds on the number of multicast addresses in use, devices shall use at most the low-order 10 bits of the Host Id in generating the range of multicast addresses. This allows for 1024 unique Host Ids.

   The following pseudo-code shows the algorithm to determine the device's starting and ending multicast addresses:

```
CIP_Mcast_Base_Addr = 0xEFC00100 // 239.192.1.0 is the starting address
CIP_Host_Mask = 0x3FF            // 10 bits of host id

if Subnet_mask configured then
   Netmask = Subnet_mask
else
   if IP_address is Class A then
      Netmask = 255.0.0.0
   else if IP_address is Class B then
      Netmask = 255.255.0.0
   else if IP_address is Class C then
      Netmask = 255.255.255.0
end_else

Host_id = IP_addr & (~Netmask)
Mcast_index = Host_id - 1
Mcast_index = Mcast_index & (CIP_Host_Mask)

Mcast_start_addr = CIP_Mcast_Base_Addr + (Mcast_index * 32)
Mcast_end_addr = Mcast_start_addr + 31
```

The following table shows example multicast assignments.

Subnet mask	IP address	Multicast addresses
None configured (8 bits of Host Id used, since 192.168.x.x is Class C)	192.168.1.1	239.192.1.0 - 239.192.1.31
	192.168.1.2	239.192.1.32 - 239.192.1.63
	192.168.1.3	239.192.1.64 - 239.192.1.95
255.255.248.0 (11 bits of Host Id)	10.10.16.1	239.192.1.0 - 239.192.1.31
	10.10.16.2	239.192.1.32 - 239.192.1.63
	10.10.16.3	239.192.1.64 - 239.192.1.95
	10.10.20.0 (Host Id = 1024; lower 10 bits all 0)	239.192.128.224 - 239.192.128.255 (this is the highest multicast address range that would result using the algorithm)

Since there are a finite number of unique Host Ids, it is possible for different devices to produce data using the same multicast address. Consequently, devices that receive packets on EtherNet/IP multicast connections shall screen the incoming packets based on the CIP Connection Id as well as the IP address of the sending device. This is described in Chapter 3-4.2.

Note that when the device's IP address or subnet mask changes, the IP multicast addresses generated by the algorithm also shall change accordingly.

2. Explicit configuration of IP multicast addresses:

   IP multicast addresses are configured via the TCP/IP Interface Object (class 0xF5). The user (or software) can configure a starting multicast address and number of multicast addresses to allocate. The configured multicast addresses shall then be used for EtherNet/IP multicast packets.

EtherNet/IP devices shall at least implement the algorithmic method for allocating IP multicast addresses, and are encouraged to implement both methods. If both methods are implemented, the algorithmic method shall be the default "out of box" method.

3-5.4 User Considerations (informative)

This section is informational, and is meant to be an aid to vendors and users in the practical deployment of EtherNet/IP applications. When deploying an EtherNet/IP system that uses multicast connections, the user should consider a number of aspects in order to achieve satisfactory application performance.

1. When devices allocate IP multicast addresses according to the default algorithm in section 3-6.3, the assignment of IP addresses to devices affects the way that IP multicast addresses are selected. Users should be aware that only 10 bits the IP address are used to generate the Host Id, which in turn determines the device's range of IP addresses. If the user's subnet mask is larger than 10 bits, there is the potential for multiple devices to use the same IP multicast address when producing data. While this does not result in incorrect operation, it can result in devices experiencing performance degradation due to the receipt of additional multicast packets that must be discarded.

2. When multicast addresses are explicitly configured, care should be taken so that devices in the same subnet have unique blocks of multicast addresses. Further, if multicast connections will cross subnet boundaries, then care must be taken to ensure that all devices in the network have unique blocks of multicast addresses. When configuring multicast addresses, it is recommended that addresses from the IPv4 Local Scope be used (239.255.0.0 – 239.255.255.255) so as not to conflict with multicast address that may be generated algorithmically.

3. Some routers experience performance degradation when they must handle many multicast packets with TTL equal to 1. In such situations, users may configure TTL to be greater than 1 even though I/O connections do not need to cross subnets. When setting TTL greater than 1, it is recommended that users also configure multicast addresses for each device. If multicast addresses are not explicitly configured, they are generated according to the algorithm in the specification. Care must be taken in this situation since devices in different subnets could generate the same IP multicast addresses. The multicast packets sent from one subnet would then be received on the other subnet, possibly impacting performance. In order to prevent unwanted multicast propagation, the user must perform additional router configuration to constrain the EtherNet/IP multicast packets to the subnet on which they originate. There are several techniques for constraining multicast at the router. Router configuration is beyond the scope of this specification.

4. Users are strongly recommended to use switches that implement IGMP snooping. When IGMP snooping is used, devices will only receive the multicast packets in which they are interested (i.e., for which they have issued an IGMP membership message).

3-5.5 Future Directions for EtherNet/IP (informative)

There are a number of Internet standards regarding IP multicast allocation and scoping. While these standards have not yet been widely deployed, they are expected to have an impact on future EtherNet/IP mechanisms for using IP multicast. Three RFC's that seem most relevant to EtherNet/IP are listed below:

- "The Internet Multicast Address Allocation Architecture", RFC 2908
- "Administratively Scoped Multicast", RFC 2365
- "Multicast Address Dynamic Client Allocation Protocol (MADCAP)", RFC 2730

3-6 IGMP Usage

3-6.1 Background (informative)

The Internet Group Management Protocol (IGMP) is a standard protocol used by hosts to report their IP multicast group memberships and must be implemented by any host that wishes to receive IP multicast datagrams. IGMP messages are used by multicast routers to learn which multicast groups have members on their attached networks. IGMP messages are also used by switches capable of supporting "IGMP snooping" whereby the switch listens to IGMP messages and only sends the multicast packets to ports that have joined the multicast group.

There are three versions of IGMP:

- IGMP V1 is defined in RFC1112.
- IGMP V2 is defined in RFC2236.
- IGMP V3 is defined in RFC3376.

RFC2236 and RFC3376 discuss host and router requirements for interoperation with older IGMP versions.

Since EtherNet/IP devices make extensive use of IP multicast for CIP transport class 0 and 1 connections, consistent IGMP usage by EtherNet/IP devices is essential in order to create well-functioning EtherNet/IP application networks.

3-6.2 IGMP Membership Report Messages

EtherNet/IP devices shall issue a Membership Report message (V1, V2 or V3 as appropriate) when opening a CIP connection on which they will receive multicast packets. Specifically, devices shall adhere to the following behavior:

1. When the T->O Connection Type is multicast (originator is multicast consumer), the originator shall issue a Membership Report upon receipt of a successful Forward_Open_reply. The Membership Report shall include the IP multicast address as communicated in the Forward_Open_reply.

2. When the O->T Connection Type is multicast (target is multicast consumer), the target shall issue a Membership Report upon sending a successful Forward_Open_reply. The Membership Report shall include the IP multicast address as communicated in the Forward_Open.

If the device has already issued a Membership Report for the IP multicast address (e.g., if the multicast address is being used with an existing connection) the device may, but is not required to, issue another Membership Report.

Devices shall also send Membership Report messages in response to Membership Query messages, per the IGMP RFCs.

3-6.3 IGMP Leave Group messages

Devices that support IGMP V2 shall issue a Leave Group when all the CIP connections associated with a consuming IP multicast address have either closed or timed out. Specifically, devices shall adhere to the following behavior:

1. When the T->O Connection Type is multicast (originator is multicast consumer), the originator shall issue a Leave Group upon receipt of a successful Forward_Close reply if the originator has no other open connections consuming on that IP multicast address.

2. When the O->T Connection Type is multicast (target is multicast consumer), the target shall issue a Leave Group upon sending a successful Forward_Close reply if the target has no other open connections consuming on that IP multicast address.

3. In the event of a connection timeout, the multicast consumer (whether target or originator) shall issue a Leave Group message if the multicast consumer has no other connections consuming on that IP multicast address.

3-7 Quality of Service (QoS) for EtherNet/IP Messages

3-7.1 Overview of QoS for EtherNet/IP

Quality of Service (QoS) is a general term for mechanisms that treat traffic streams with different relative priorities or other delivery characteristics. Standard QoS mechanisms include IEEE 802.1D/Q (Ethernet frame priority) and Differentiated Services (DiffServ) in the TCP/IP protocol suite.

Within the CIP and EtherNet/IP application context, QoS is especially important for time sensitive applications such as CIP Sync and CIP Motion where packet delivery will affect application stability.

EtherNet/IP defines requirements and recommendations for how EtherNet/IP devices use two standard QoS mechanisms: 802.1D/Q and Differentiated Services. The following summarizes the QoS mechanisms defined for EtherNet/IP devices:

- The overall approach is for EtherNet/IP end devices to mark EtherNet/IP packets with priority values, using the standard mechanisms mentioned above. Marking packets with priority enables infrastructure devices (e.g., switches and routers) to better differentiate EtherNet/IP traffic.
- For CIP transport class 0 and 1 connections (i.e., UDP-based), there is a defined mapping of CIP priorities to 802.1D priorities and DiffServ Code Points (see section 3-7.5).
- For UCMM and CIP transport class 3 connections (i.e., TCP-based), there is a defined DiffServ Code Point, and a defined 802.1D priority value.
- For PTP (IEEE 1588) messages, there are DiffServ Code Points and 802.1D priority values corresponding to the two different types of PTP messages.
- It is strongly recommended that devices by default mark CIP and PTP messages with DSCP values.
- In addition to DSCP, devices may optionally support sending and receiving 802.1Q tagged frames. If supported, sending tagged frames shall be disabled by default in order to prevent device interoperability problems. When 802.1Q tagging is desired, the end user must explicitly enable the sending of 802.1Q frames, and must ensure that both the sending and receiving devices support tagged frames.
- The QoS Object provides a means to configure DSCP values, and a means to enable/disable sending of 802.1Q tagged frames (see Volume 2, Chapter 5).
- There are no requirements for devices to mark traffic other than CIP or IEEE 1588. Devices may do so, depending on their implementation capabilities.
- At present there are no specific implementation requirements for end devices to internally differentiate traffic with different priorities. Traffic differentiation in end devices is an area of future development. EtherNet/IP implementations are at a minimum recommended to give higher priority to processing EtherNet/IP implicit messages over explicit messages. Further differentiation based on the above QoS mechanisms, where possible, is also recommended.

3-7.2 QoS References

The following list shows the references applicable to QoS for EtherNet/IP devices

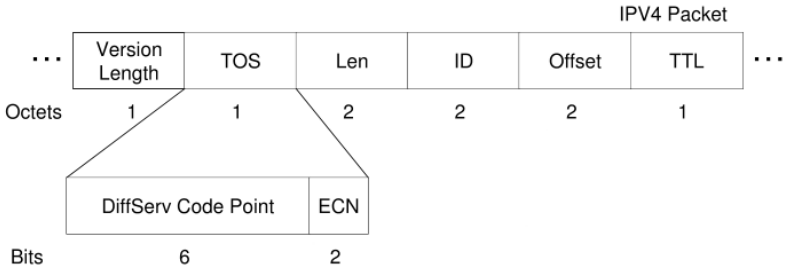
- RFC 2474 – Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
- RFC 2475 – An Architecture for Differentiated Services
- RFC 2597 – Assured Forwarding PHB Group
- RFC 3140 – Per Hop Behavior Identification Codes
- RFC 3246 – An Expedited Forwarding PHB (Per-Hop Behavior)
- RFC 4594 – Configuration Guidelines for DiffServ Service Classes
- IEEE Std 802.1D – 2004 (defines the use of priority in the 802.1Q frame format)
- IEEE Std 802.1Q – 2005 (defines VLAN operation including the tagged frame format)

3-7.3 DSCP Format

Differentiated Services (DiffServ) is a model for specifying the relative priority of traffic based on the type of service (ToS) field of an IPv4 packet. The model is defined in RFC 2475. DiffServ allows nodes to route packets based on class of traffic as defined by the DiffServ Codepoint (DSCP) and the defined Per-Hop Behavior (PHB) characteristics.

Figure 3-7.1 shows the DS field in the IP header.

Figure 3-7.1 DS Field in the IP Header



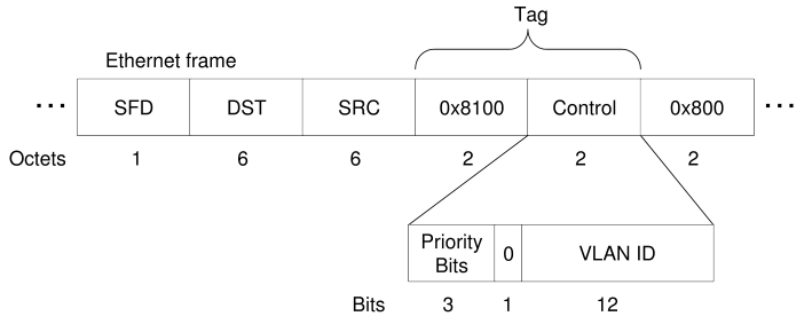
Note that when setting the DSCP value in the IP header, if placed directly in the ToS field, it must be shifted left 2 bits.

3-7.4 IEEE 802.1D/Q format

IEEE 802.1Q defines an Ethernet frame format that allows inclusion of VLAN ID and priority. The 802.1Q frame has EtherType of 0x8100 and a 4-byte prefix between the Source and Type fields of the frame. The tagged frame defines a 3-bit field to specify 8 priority levels, further specified in 802.1D. Priority 7 is the highest. Priority 0 is the lowest.

Figure 3-7.2 shows the format of an 802.1Q frame.

Figure 3-7.2 802.1Q Tagged Frame



3-7.5 Mapping CIP Traffic to DSCP and 802.1D

Table 3-7.1 defines the default DSCP and 802.1D priority mappings for EtherNet/IP and CIP Sync (IEEE 1588) traffic.

Table 3-7.1 Default DSCP and 802.1D Mapping for EtherNet/IP

Traffic Type	CIP Priority	DSCP	802.1D Priority[1]	CIP Traffic Usage (recommended)
PTP Event (IEEE 1588)	n/a	59 (‘111011’)	7	n/a
PTP General (IEEE 1588)	n/a	47 (‘101111’)	5	n/a
CIP class 0 / 1	Urgent (3)	55 (‘110111’)	6	CIP Motion
	Scheduled (2)	47 (‘101111’)	5	Safety I/O I/O
	High (1)	43 (‘101011’)	5	I/O
	Low (0)	31 (‘011111’)	3	No recommendation at present
CIP UCMM CIP class 3 All other EtherNet/IP encapsulation messages	All	27 (‘011011’)	3	CIP messaging

1 Sending 802.1Q tagged frames is disabled by default

In Table 3-7.1 above, note that the 802.1D and DSCP values for transport class 0 and class 1 messages are based on the CIP priority (indicated in the Forward Open service). PTP and CIP explicit messages use 802.1D and DSCP values independent of CIP priority.

The default DSCP values and 802.1Q tagged frame enable/disable may be changed via the QoS Object (see chapter 5).

Note: The default DSCP values are “local use” values, as defined in RFC 2474.

3-7.6 EtherNet/IP usage of DSCP

When marking EtherNet/IP traffic with DSCP values, EtherNet/IP devices shall use the values as configured in the QoS Object (default values are shown in Table 3-7.1). Usage of the ECN field in the IP header by EtherNet/IP end devices is not currently defined and shall be set to 0.

Devices are strongly encouraged to support marking EtherNet/IP packets with DSCP values. When supported, the default behavior shall be to mark packets.

All EtherNet/IP devices shall support receiving packets with non-zero DSCP values since this is an Internet Protocol requirement (see RFC 791 and RFC 1122). Note that Ethernet infrastructure devices (e.g., switches and routers) can potentially alter DSCP values. Receiving devices shall not assume or otherwise check that incoming DSCP values are the same as in Table 3-7.1 above, or are the same values as sent from the sending device.

3-7.7 EtherNet/IP usage of 802.1D/Q

When sending traffic with 802.1Q tagged frames, EtherNet/IP devices shall use the priority values specified in Table 3-7.1. The VLAN ID shall be set to 0, unless a specific VLAN ID for the device has been configured by means not covered by this specification. When receiving 802.1Q tagged frames, devices shall not require that the VLAN ID be set to 0.

When sending 802.1Q tagged frames, devices shall also set the corresponding DSCP value in the IP header.

When 802.1Q frames are supported, the device shall also support the QoS Object (see Chapter 5). The default behavior shall be to disable sending of 802.1Q tagged frames, since sending tagged frames by default can result in device interoperability problems.

3-7.8 User considerations with 802.1D/Q

Some EtherNet/IP devices do not support receiving 802.1Q tagged frames. When 802.1Q frames are sent to such devices, the frames will be dropped. In addition, some managed switches will drop 802.1Q tagged frames unless the receiving port is configured to accept them.

In order to prevent interoperability problems, sending 802.1Q frames is disabled by default for EtherNet/IP devices. The end user may elect to enable sending tagged frames when supported, via the QoS Object. The user is ultimately responsible for ensuring that both the sending and receiving devices support 802.1Q frames, and that network infrastructure is properly configured.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 4: Object Model

Contents

4-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the CIP object model that are EtherNet/IP specific. At this time, no such additions exist.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 5: Object Library

Contents

5-1 Introduction

In this standard, object modeling is used to represent the network visible behavior of devices. Devices are modeled as a collection of objects. Each class of objects is a collection of related services, attributes and behaviors. Services are the procedures that an object performs. Attributes are characteristics of objects represented by values, which can vary. An object's behavior is an indication of how the object responds to particular events.

This chapter of the specification contains the object descriptions specific to EtherNet/IP. The rest of the object descriptions can be found in the Volume 1, Chapter 5. With respect to the OSI reference model, CIP objects perform the Layer 7 Application functions. They also provide a mechanism to access station management counters via the network.

5-2 Reserved Class Codes

The rest of the class codes are defined in Volume 1 of the CIP Networks Library.

5-3 TCP/IP Interface Object

Class Code: F5 Hex

5-3.1 Scope

The TCP/IP Interface Object provides the mechanism to configure a device's TCP/IP network interface. Examples of configurable items include the device's IP Address, Network Mask, and Gateway Address.

The underlying physical communications interface associated with the TCP/IP Interface Object shall be any interface that supports the TCP/IP protocol. For example, a TCP/IP Interface Object may be associated any of the following: an IEEE 802.3 interface, an ATM interface, a serial port running SLIP, a serial port running PPP, etc. The TCP/IP Interface Object provides an attribute that identifies the link-specific object for the associated physical communications interface. The link-specific object is generally expected to provide link-specific counters as well as any link-specific configuration attributes.

Each device shall support exactly one instance of the TCP/IP Interface Object for each TCP/IP-capable communications interface on the module.

5-3.2 Revision History

Since the initial release of this object class definition changes have been made that require a revision update of this object class. The table below represents the revision history.

Table 5-3.1 Revision History

Revision	Reason for Object Definition Update:
1	Initial revision of this object definition
2	Added ACD Instance attributes 10 (SelectACD) and 11 (LastConflictDetected) Added Instance Attribute 12, EtherNet/IP Quick Connect Added bits 5 (Interface Configuration Pending) and 6 (AcdStatus) to Attribute 1, Status Added bits 6 (Interface Configuration Change Requires Reset) and 7 (AcdCapable) to Attribute 2, Configuration Capability

5-3.3 Attributes

5-3.3.1 Class Attributes

Table 5-3.2 Class Attributes

Attr ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of values
1	Required	Get	NV	Revision	UINT	Revision of this object	The value shall be 2.
2	Conditional [1]	Get	NV	Max Instance	UINT	Maximum instance number of an object currently created in this class level of the device.	The largest instance number of a created object at this class hierarchy level.
3	Conditional [1]	Get	NV	Number of Instances	UINT	Number of object instances currently created at this class level of the device.	The number of object instances at this class hierarchy level
4 thru 7	These class attributes are optional and are described in Volume 1, Chapter 4.						

1 Required if the number of instances is greater than 1.

5-3.3.2 Instance Attributes

Table 5-3.3 Instance Attributes

Attr ID	Need In Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
1	Required	Get	V	Status	DWORD	Interface status	See section 5-3.3.2.1.
2	Required	Get	NV	Configuration Capability	DWORD	Interface capability flags	Bit map of capability flags. See section 5-3.3.2.2.
3	Required	Get Set is condi-tional [1]	NV	Configuration Control	DWORD	Interface control flags	Bit map of control flags. See section 5-3.3.2.3
4	Required	Get	NV	Physical Link Object	STRUCT of:	Path to physical link object	See section 5-3.3.2.4
				Path size	UINT	Size of Path	Number of 16 bit words in Path
				Path	Padded EPATH	Logical segments identifying the physical link object	The path is restricted to one logical class segment and one logical instance segment. The maximum size is 12 bytes. See Appendix C of Volume 1, Logical Segments.

TCP/IP Object, Class Code: $F5_{Hex}$

Attr ID	Need In Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
5	Required	Get Set is recom-mended	NV when Configuration Method is 0. V when obtained via BOOTP or DHCP	Interface Configuration	STRUCT of:	TCP/IP network interface configuration.	See section 5-3.3.2.5
				IP Address	UDINT	The device's IP address.	Value of 0 indicates no IP address has been configured. Otherwise, the IP address shall be set to a valid Class A, B, or C address and shall not be set to the loopback address (127.0.0.1).
				Network Mask	UDINT	The device's network mask	Value of 0 indicates no network mask address has been configured.
				Gateway Address	UDINT	Default gateway address	Value of 0 indicates no IP address has been configured. Otherwise, the IP address shall be set to a valid Class A, B, or C address and shall not be set to the loopback address (127.0.0.1).
				Name Server	UDINT	Primary name server	Value of 0 indicates no name server address has been configured. Otherwise, the name server address shall be set to a valid Class A, B, or C address.
				Name Server 2	UDINT	Secondary name server	Value of 0 indicates no secondary name server address has been configured. Otherwise, the name server address shall be set to a valid Class A, B, or C address.
				Domain Name	STRING	Default domain name	ASCII characters. Maximum length is 48 characters. Shall be padded to an even number of characters (pad not included in length). A length of 0 shall indicate no Domain Name is configured.

TCP/IP Object, Class Code: $F5_{Hex}$

Attr ID	Need In Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
6	Required	Get Set is Condi-tional [2]	NV	Host Name	STRING	Host name	ASCII characters. Maximum length is 64 characters. Shall be padded to an even number of characters (pad not included in length). A length of 0 shall indicate no Host Name is configured. See section 5-3.3.2.6.
7	Conditional [3]			Safety Network Number	6 octets	See CIP Safety Specfication, Volume 5, Chapter 3	
8	Conditional [4]	Get Set is condi-tional [5]	NV	TTL Value	USINT	TTL value for EtherNet/IP multicast packets	Time-to-Live value for IP multicast packets. Default value is 1. Minimum is 1; maximum is 255 See Chapter 5-3.3.2.7.
9	Conditional [4]	Get Set is condi-tional [5]	NV	Mcast Config	STRUCT of:	IP multicast address configuration	See Chapter 5-3.3.2.8.
				Alloc Control	USINT	Multicast address allocation control word. Determines how addresses are allocated.	See Chapter 5-3.3.2.8 for details. Determines whether multicast addresses are generated via algorithm or are explicitly set.
				Reserved	USINT	Reserved for future use	Shall be 0.
				Num Mcast	UINT	Number of IP multicast addresses to allocate for EtherNet/IP	The number of IP multicast addresses allocated, starting at "Mcast Start Addr". Maximum value is device specific, however shall not exceed the number of EtherNet/IP multicast connections supported by the device.
				Mcast Start Addr	UDINT	Starting multicast address from which to begin allocation.	IP multicast address (Class D). A block of "Num Mcast" addresses is allocated starting with this address.

TCP/IP Object, Class Code: $F5_{Hex}$

Attr ID	Need In Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
10	Conditional [6]	Set	NV	SelectAcd	BOOL	Activates the use of ACD	Enable ACD (1, default), Disable ACD (0). See section 5-3.3.2.9
11	Conditional [6]	Set	NV	LastConflictDetected	STRUCT of:	Structure containing information related to the last conflict detected	ACD Diagnostic Parameters See section 5-3.3.2.10
				AcdActivity	USINT	State of ACD activity when last conflict detected	ACD activity Default = 0 See section 5-3.3.2.10
				RemoteMAC	Array of 6 USINT	MAC address of remote node from the ARP PDU in which a conflict was detected	MAC from Eth Pkt Hdr. Default =0 See section 5-3.3.2.10
				ArpPdu	ARRAY of 28 USINT	Copy of the raw ARP PDU in which a conflict was detected.	ARP PDU Default = 0 See section 5-3.3.2.10
12	Optional	Set	NV	EtherNet/IP Quick_Connect	BOOL	Enable/Disable of Quick Connect feature	0 = Disable (default) 1 = Enable See Section 5-3.3.2.11. For information regarding Quick Connect, refer to Appendix E – EtherNet/IP Quick Connect

1 Set is required unless the configuration method is selected exclusively via hardware setting.

2 The set access is optional when the Interface Configuration atribute is not settable.

3 This attribute is required for EtherNet/IP safety devices. Non-safety devices shall not implement this attribute.

4 If either TTL Value or Mcast Config is implemented, both must be implemented.

5 If either TTL Value or Mcast Config is implemented as settable, both must be implemented as settable.

6 REQUIRED if device implements ACD.

For information regarding Quick Connect, refer to Appendix E – EtherNet/IP Quick Connect.

For information regarding ACD refer to Appendix F – Address Conflict Detection.

5-3.3.2.1 Status Instance Attribute

The **Status** attribute is a bitmap that shall indicate the status of the TCP/IP network interface. Refer to the state diagram in section 5-3.5, Behavior, for a description of object states as they relate to the Status attribute.

Table 5-3.4 Status Attribute

Bit(s):	Called:	Definition	
0-3	Interface Configuration Status	Indicates the status of the Interface Configuration attribute.	0 = The Interface Configuration attribute has not been configured. 1 = The Interface Configuration attribute contains valid configuration obtained from BOOTP, DHCP or non-volatile storage. 2 = The IP address member of the Interface Configuration attribute contains valid configuration, obtained from hardware settings (e.g.: pushwheel, thumbwheel, etc.) 3-15 = Reserved for future use.
4	Mcast Pending	Indicates a pending configuration change in the TTL Value and/or Mcast Config attributes. This bit shall be set when either the TTL Value or Mcast Config attribute is set, and shall be cleared the next time the device starts.	
5	Interface Configuration Pending	Indicates a pending configuration change in the Interface Configuration attribute. This bit shall be 1 (TRUE) when Interface Configuration attribute are set and the device requires a reset in order for the configuration change to take effect (as indicated in the Configuration Capability attribute). The intent of the Interface Config Pending bit is to allow client software to detect that a device's IP configuration has changed, but will not take effect until the device is reset.	
6	AcdStatus	Set (1) Address Conflict Detected, Clear (0) No Address Conflict Detected	
7-31	Reserved	Reserved for future use and shall be set to zero.	

5-3.3.2.2 Configuration Capability Instance Attribute

The Configuration Capability attribute is a bitmap that indicates the device's support for optional network configuration capability. Devices are not required to support any one particular item, however must support at least one method of obtaining an initial IP address.

Table 5-3.5 Configuration Capability Attribute

Bit(s):	Called:	Definition
0	BOOTP Client	1 (TRUE) shall indicate the device is capable of obtaining its network configuration via BOOTP.
1	DNS Client	1 (TRUE) shall indicate the device is capable of resolving host names by querying a DNS server.
2	DHCP Client	1 (TRUE) shall indicate the device is capable of obtaining its network configuration via DHCP.
3	DHCP-DNS Update	Shall be 0, behavior to be defined in a future specification edition.
4	Configuration Settable	1 (TRUE) shall indicate the Interface Configuration attribute is settable.
5	Hardware Configurable	1 (TRUE) shall indicate the IP Address member of the Interface Configuration attribute can be obtained from hardware settings (e.g., pushwheel, thumbwheel, etc.). If this bit is FALSE the Status Instance Attribute (1), Interface Configuration Status field value shall never be 2 (The Interface Configuration attribute contains valid configuration, obtained from hardware settings)
6	Interface Configuration Change Requires Reset	1 (TRUE) shall indicate that the device requires a restart in order for a change to the Interface Configuration attribute to take effect. If this bit is FALSE a change in the Interface Configuration attribute will take effect immediately.
7	AcdCapable	(1) TRUE shall indicate that the device is ACD capable
8-31	Reserved	Reserved for future use and shall be set to zero.

5-3.3.2.3 Configuration Control Instance Attribute

5-3.3.2.3.1 Configuration Control Structure

The **Configuration Control** attribute is a bitmap used to control network configuration options. Table 5-3.6 shows the structure of the Configuration Control attribute:

Table 5-3.6 Configuration Control Attribute

Bit(s):	Called:	Definition	
0-3	Configuration Method	Determines how the device shall obtain its IP-related configuration	0 = The device shall use statically-assigned IP configuration values. 1 = The device shall obtain its interface configuration values via BOOTP. 2 = The device shall obtain its interface configuration values via DHCP. 3-15 = Reserved for future use.
4	DNS Enable	If 1 (TRUE), the device shall resolve host names by querying a DNS server.	
5-31	Reserved	Reserved for future use and shall be set to zero.	

5-3.3.2.3.2 Configuration Method

The Configuration Method determines how a device shall obtain its IP-related configuration:

- If the Configuration Method is 0, the device shall use statically-assigned IP configuration contained in the Interface Configuration attribute (or assigned via non-CIP methods, as noted below).

- If the Configuration Method is 1, the device shall obtain its IP configuration via BOOTP. The BOOTP client behavior shall be as defined in the relevant RFCs (RFC 951, RFC 1542, RFC 2132) or their successors.
- If the Configuration Method is 2, the device shall obtain its IP configuration via DHCP. The DHCP client behavior shall be as defined in the relevant RFCs (RFC 2131, RFC 2132) or their successors.
- Devices that optionally provide hardware means (e.g., rotary switch) to configure IP addressing behavior shall set the Configuration Method to reflect the configuration set via hardware: 0 if a static IP address has been configured, 1 if BOOTP has been configured, 2 if DHCP has been configured.

If a device has been configured to obtain its configuration via BOOTP or DHCP it shall continue sending requests until a response from the server is received. Devices that elect to use default IP configuration in the event of no response from the server shall continue issuing requests until a response is received, or until the Configuration Method is changed to static.

Once the device receives a response from the server it shall stop sending the BOOTP/DHCP client requests (DHCP clients shall follow the lease renewal behavior per the RFC). It is recommended that devices implement the means to detect a link up and upon a link up detection restart the initial BOOTP or DHCP sequence. For multiport devices the restart of the initial BOOTP or DHCP sequence shall only be triggered if all external links have been down and when the first link up is detected.

Setting the Configuration Method to 0 (static address) shall cause the Interface Configuration to be saved to NV storage.

It is recommended that setting the Configuration Method to 1 (BOOTP) or 2 (DHCP) cause the device to start the BOOTP / DHCP client to obtain new IP address configuration. If the device requires a reset in order to start the BOOTP / DHCP client, it shall set the Interface Config Pending bit, and upon device reset start the BOOTP / DHCP client.

5-3.3.2.3.3 DNS Enable

For originator devices that support resolving target host names via DNS, the DNS Enable bit shall enable (1) and disable (0) the DNS client.

5-3.3.2.4 Physical Link Object

This attribute identifies the object associated with the underlying physical communications interface (e.g., an 802.3 interface). There are two components to the attribute: a Path Size (in UINTs) and a Path. The Path shall contain a Logical Segment, type Class, and a Logical Segment, type Instance that identifies the physical link object. The maximum Path Size is 6 (assuming a 32 bit logical segment for each of the class and instance).

The physical link object itself typically maintains link-specific counters as well as any link-specific configuration attributes. If the CIP port associated with the TCP/IP Interface Object has an Ethernet physical layer, this attribute shall point to an instance of the Ethernet Link Object (class code = 0xF6). When there are multiple physical interfaces that correspond to the TCP/IP interface, this attribute shall either contain a Path Size of 0, or shall contain a path to the object representing an internal communications interface (often used in the case of an embedded switch).

For example, the path could be as follows:

Table 5-3.7 Example Path

Path	Meaning
[20][F6][24][01]	[20] = 8 bit class segment type; [F6] = Ethernet Link Object class; [24] = 8 bit instance segment type; [01] = instance 1.

5-3.3.2.5 Interface Configuration

5-3.3.2.5.1 Interface Configuration Contents

The Interface Configuration attribute contains the configuration parameters required for a device to operate as a TCP/IP node. The contents of the Interface Configuration attribute shall depend upon how the device has been configured to obtain its IP parameters:

- If configured to use a static IP address (Configuration Method value is 0), the Interface Configuration values shall be those which have been statically assigned and stored in NV storage.
- If configured to use BOOTP or DHCP (Configuration Method value is 1 or 2), the Interface Configuration values shall contain the configuration obtained from the BOOTP or DHCP server. The Interface Configuration attribute shall be 0 until the BOOTP/DHCP reply is received.
- Some devices optionally provide additional, non-CIP mechanisms for setting IP-related configuration (e.g., a web server interface, rotary switch for configuring IP address, etc.). When such a mechanism is used, the Interface Configuration attribute shall reflect the IP configuration values in use.

Table 5-3.8 Interface Configuration Attribute

Name	Meaning
IP address	The device's IP address.
Network mask	The device's network mask. The network mask is used when the IP network has been partitioned into subnets. The network mask is used to determine whether an IP address is located on another subnet.
Gateway address	The IP address of the device's default gateway. When a destination IP address is on a different subnet, packets are forwarded to the default gateway for routing to the destination subnet.
Name server	The IP address of the primary name server. The name server is used to resolve host names. For example, that might be contained in a CIP connection path.
Name server 2	The IP address of the secondary name server. The secondary name server is used when the primary name server is not available, or is unable to resolve a host name.
Domain name	The default domain name. The default domain name is used when resolving host names that are not fully qualified. For example, if the default domain name is "odva.org", and the device needs to resolve a host name of "plc", then the device will attempt to resolve the host name as "plc.odva.org".

For additional information on IP addressing, subnetworks, gateways, etc. refer to Comer, Douglas E.*; Internetworking with TCP/IP, Volume 1: Protocols and Architecture*; Englewood Cliffs, NJ; Prentice-Hall, 1990.

5-3.3.2.5.2 Set Attributes Behavior

In order to prevent incomplete or incompatible configuration, the parameters making up the Interface Configuration attribute cannot be set individually. To modify the Interface Configuration attribute, client software should first Get the Interface Configuration attribute, change the desired parameters, and then Set the attribute.

An attempt to set any of the parameters of the Interface Configuration attribute to invalid values (see Semantics of Values in Table 5-3.2) shall result in an error response with status code 0x09 'Invalid Attribute Value' to be returned. In this scenario, all of the parameters of the Interface Configuration attribute retain the values that existed prior to the invocation of the set service.

If the device has an active I/O connection, it is recommended that the device rejects the set attributes request by returning an error response with status code 0x10 'Device State Conflict'.

When the value of the Configuration Method (Configuration Control attribute) is 0, the set attribute service shall store the new Interface Configuration values in non-volatile memory. If the device requires reset in order for new parameters to take effect (Configuration Capability bit 6 set), the device shall set the Interface Configuration Pending bit (Status attribute bit 5).

After storing the new values the device shall send a response using its current IP address (i.e., the IP address to which the set attributes request was sent).

If the device does not require reset (Configuration Capability bit 6 clear) in order for new IP configuration to take effect, after responding to the set service the device shall initiate application of the new IP parameters. While implementation-dependent, this activity is generally initiated by the TCP/IP Interface Object set service and then completed asynchronously by the TCP/IP stack.

In order to achieve consistency of device configuration behavior, it is recommended that devices support setting the Interface Configuration attribute, and support application of new attribute values without requiring device reset. If a device does not support setting the Interface Configuration attribute, the device shall return an error response with status code 0x0E 'Attribute Not Settable'.

5-3.3.2.5.3 Application of New Configuration Parameters

If the device does not require reset (Configuration Capability bit 6 clear), after responding to the set service the device shall initiate application of the new IP parameters. When applying new IP configuration, the device shall maintain the consistency of the IP configuration context in which it operates. For example, the device shall not mix old and new IP address / network mask / gateway address values, and must not use the new IP configuration on CIP connections established with the previous IP address.

When a TCP/IP stack is configured to use a particular set of IP parameters, a context around these IP parameters is built. This context includes relationships to the TCP/IP stack, the CIP communications environment, and the control application among other entities. To allow a different set of IP parameters to be used a new IP context shall be built. The device shall properly manage its IP context and maintain its consistency. For example a new IP Address shall not be used with old Network Mask or Gateway Address; don't use old IP address for CIP or other communication once the new one is applied.

5-3.3.2.6 Host Name

The **Host Name** attribute contains the device's host name, which can be used for informational purposes. The set access is optional when the Interface Configuration attribute is not settable.

5-3.3.2.7 TTL Value

TTL Value is value the device shall use for the IP header Time-to-Live field when sending EtherNet/IP packets via IP multicast. By default, TTL Value shall be 1. The maximum value for TTL is 255. Note that unicast packets shall use the TTL as configured for the TCP/IP stack, and not the TTL Value configured in this attribute.

When set, the TTL Value attribute shall be saved in non-volatile memory. If a device does not support applying the TTL Value immediately, the Mcast Pending bit in the Interface Status attribute shall be set, indicating that there is pending configuration. For devices that support applying the TTL Value immediately, if there are existing multicast connections, an Object State Conflict error (0xC) shall be returned and the Mcast Pending bit shall not be set. When a new TTL Value is pending, Get_Attribute_Single or Get_Attributes_All requests shall return the pending value. The Mcast Pending bit shall be cleared the next time the device starts.

Users should exercise caution when setting the TTL Value greater than 1, to prevent unwanted multicast traffic from propagating through the network. Chapter 3 includes a discussion on user considerations when using multicast.

5-3.3.2.8 Mcast Config

The **Mcast Config** attribute contains the configuration of the device's IP multicast addresses to be used for EtherNet/IP multicast packets. There are three elements to the Mcast Config structure: Alloc Control, Num Mcast, and Mcast Start Addr.

Alloc Control determines how the device shall allocate IP multicast addresses (e.g., whether by algorithm, whether they are explicitly set, etc.) Table 5-3.9 shows the details for Alloc Control.

Table 5-3.9 Alloc Control

Value	Definition
0	Multicast addresses shall be generated using the default allocation algorithm specified in Chapter 3. When this value is specified on a set-attribute or set-attributes-all, the values of Num Mcast and Mcast Start Addr in the set-attribute request shall be 0.
1	Multicast addresses shall be allocated according to the values specified in Num Mcast and Mcast Start Addr.
2	Reserved

Num Mcast is the number of IP multicast addresses allocated. The maximum number of multicast addresses is device specific, but shall not exceed the number of EtherNet/IP multicast connections supported by the device.

Mcast Start Addr is the starting multicast address from which Num Mcast addresses are allocated.

When set, the Mcast Config attribute shall be saved in non-volatile memory. If a device does not support applying the MCast Config attribute immediately, the Mcast Pending bit in the Interface Status attribute shall be set, indicating that there is pending configuration. For devices that support applying the Mcast Config attribute immediately, if there are existing multicast connections an Object State Conflict error (0xC) shall be returned and the MCast Pending bit shall not be set. When a new Mcast Config value is pending, Get_Attribute_Single or Get_Attributes_All requests shall return the pending value. The Mcast Pending bit shall be cleared the next time the device starts.

When the multicast addresses are generated using the default algorithm, Num Mcast and Mcast Start Addr shall report the values generated by the algorithm.

5-3.3.2.9 SelectAcd

SelectAcd is an attribute used to Enable/Disable ACD.

If SelectAcd is 0 then ACD is disabled. If SelectAcd =1 then ACD is enabled.

The default value of SelectAcd shall be 1 indicating that ACD is enabled.

When the value of SelectAcd is changed by a Set_Attribute service, the new value of SelectAcd shall not be applied until the device executes a restart.

5-3.3.2.10 LastConflictDetected

The LastConflictDetected attribute is a diagnostic attribute presenting information about the ACD state when the last IP Address conflict was detected.

To reset this attribute the Set_Attribute_Single service is invoked with an attribute value of all 0. Values other than 0 shall result in an error response (status code 0x09, Invalid Attribute Value).

AcdActivity – The ACD contains the state of the ACD algorithm when the last IP address conflict was detected. The ACD activities are defined in the following table.

Table 5-3.10 AcdActivity

Value	AcdMode
0	NoConflictDetected (Default)
1	ProbeIpv4Address
2	OngoingDetection
3	SemiActiveProbe

RemoteMac – The IEEE 802.3 source MAC address from the header of the received Ethernet packet which was sent by a device reporting a conflict.

ArpPdu – The ARP Response PDU in binary format.

The ArpPdu shall be a copy of the ARP message that caused the address conflict. It SHALL be a raw copy of the ARP message as it appears on the Ethernet network, i.e.: ArpPdu[1] contains the first byte of the ArpPdu received.

Table 5-3.11 ArpPdu - The ARP Response PDU in binary format

Field Size [bytes]	Field Description	Field Value
2	Hardware Address Type	1 for Ethernet H/W
2	Protocol Address Type	0x800 for IP
1	HADDR LEN	6 for Ethernet h/w
1	PADDR LEN	4 for IP
2	OPERATION	1 for Req or 2 for Rsp
6	SENDER HADDR	Sender's h/w addr
4	SENDER PADDR	Sender's proto addr
6	TARGET HADDR	Target's h/w addr
4	TARGET PADDR	Target's proto addr

5-3.3.2.11 EtherNet/IP Quick_Connect

The EtherNet/IP Quick_Connect attribute shall enable (1) or disable (0) the Quick Connect feature. The default value of the attribute shall be 0.

For information regarding Quick Connect, refer to Appendix E – "EtherNet/IP Quick Connect".

5-3.4 Common Services

5-3.4.1 All Services

The TCP/IP Interface Object shall provide the following common services.

Table 5-3.12 Common Services

Service	Need in Implementation			
Code	Class	Instance	Service name	Description of Service
0x01	Optional	Optional	Get_Attributes_All	Returns a predefined listing of this objects attributes (See the Get_Attributes_All response definition in section 5-3.4.2)
0x02	n/a	Optional	Set_Attributes_All	Modifies all settable attributes.
0x0E	Required	Required	Get_Attribute_Single	Returns the contents of the specified attribute.
0x10	n/a	Required	Set_Attribute_Single	Modifies a single attribute.

5-3.4.2 Get_Attributes_All Response

At the class level, the Get_Attributes_All response shall contain the class attributes in numerical order, up to the last implemented attribute. Any unimplemented attributes in the response shall use the default attribute values.

For instance attributes, attributes shall be returned in numerical order up to the last implemented attribute. The Get_Attributes_All reply shall be as follows:

Table 5-3.13 Get_Attributes_All

Attribute ID	Size in Bytes	Contents
1	4	Status
2	4	Configuration Capability
3	4	Configuration Control
4	2	Physical Link Object, Path Size
	Variable, 12 bytes max	Physical Link Object, Path (if Path Size is non-zero)
5	4	IP Address
	4	Network Mask
	4	Gateway Address
	4	Name Server
	4	Secondary Name Server
	2	Domain Name Length
	Variable, equal to Domain Name Length	Domain Name
	1	Pad byte only if Domain Name Length is odd
6	2	Host Name Length
	Variable, equal to Host Name Length	Host Name
	1	Pad byte only if Host Name Length is odd
7	6 octets	See CIP Safety Specification Volume 5, Chapter 3. Not present if attribute 7 is not implemented. Shall be 0 when additional attributes greater than attribute ID 7 are included.
8	1 octet	TTL Value. Not present if attribute 8 is not implemented. Shall be 0 when additional attributes greater than attribute ID 8 are included.
9	8 octets	Mcast Config. Not present if attribute 9 is not implemented. Shall be 0 when additional attributes greater than attribute ID 9 are included.
10	1 octet	SelectACD value. Not present if attribute 10 is not implemented. Shall be 0 when additional attributes greater than attribute ID 10 are included.

Attribute ID	Size in Bytes	Contents
11	1 octet	AcdActivity Not present if attribute 11 is not implemented. Shall be 0 when additional attributes greater than attribute ID 11 are included.
	6 octets	RemoteMAC Not present if attribute 11 is not implemented. Shall be 0 when additional attributes greater than attribute ID 11 are included.
	28 octets	ArpPdu Not present if attribute 11 is not implemented. Shall be 0 when additional attributes greater than attribute ID 11 are included.
12	1 octet	EtherNet/IP Quick Connect Not present if attribute 12 is not implemented.

The lengths of the Physical Link Object path, Domain Name, and Host Name are not known before issuing the Get_Attributes_All service request. Implementers shall be prepared to accept a response containing the maximum sizes of the Physical Link Object path (6 UINTs), the Domain Name (48 USINTs), and Host Name (64 USINTs).

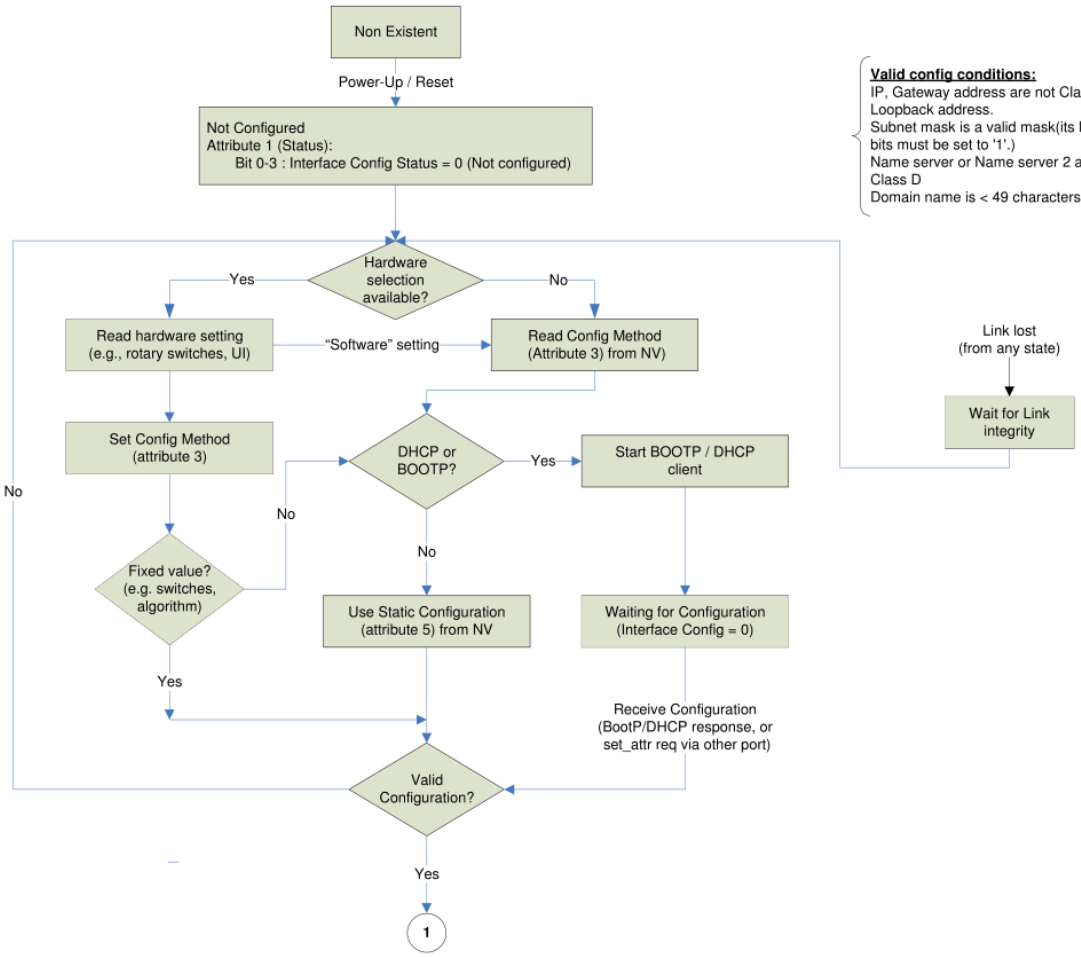
5-3.4.3 Set_Attributes_All Request

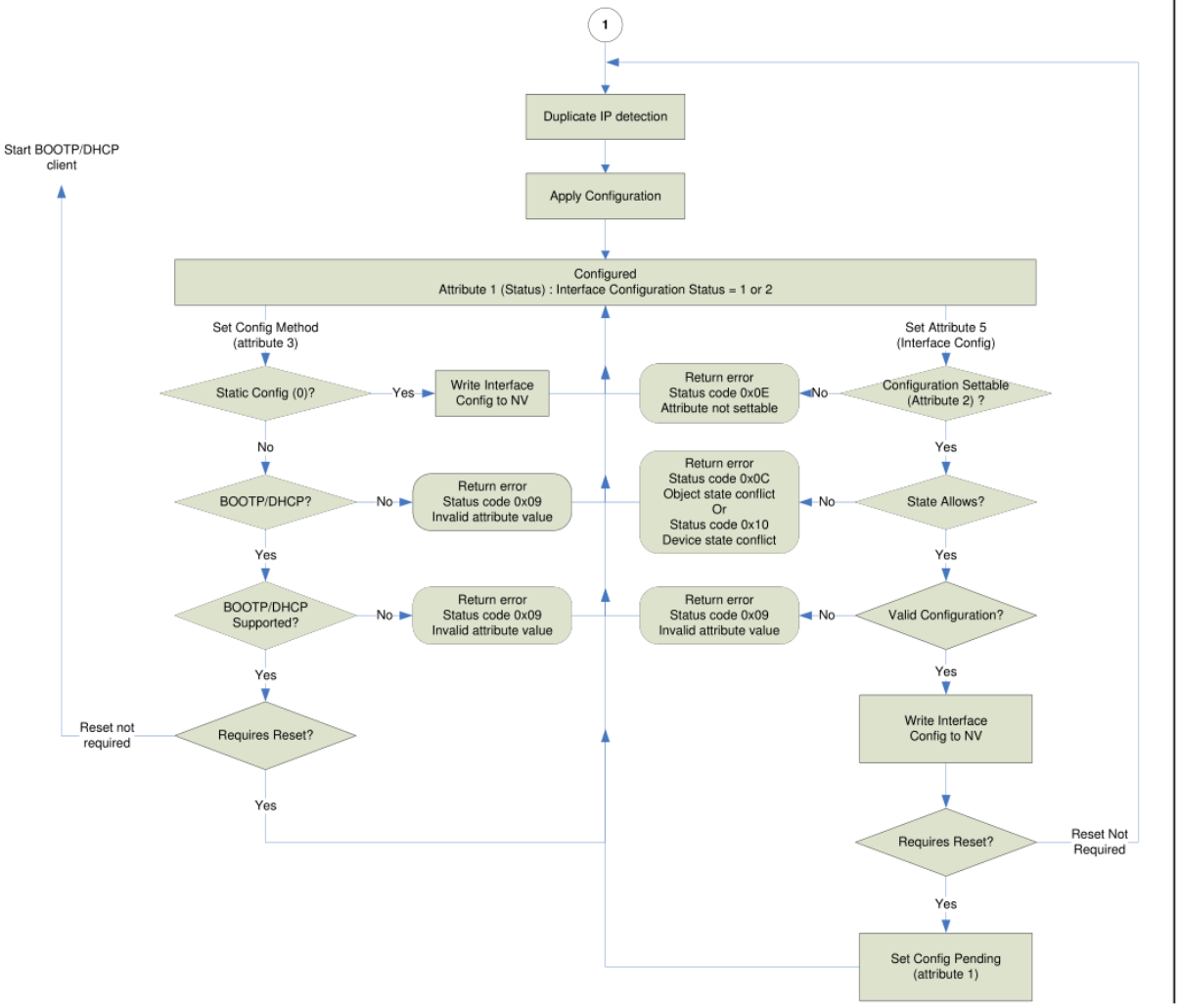
The instance Set_Attributes_All request contains the Configuration Control attribute, followed by the Interface Configuration attribute.

5-3.5 Behavior

The behavior of the TCP/IP Interface Object shall be as illustrated in the Diagram below. Note that the act of obtaining an initial executable image via BOOTP/TFTP shall not be considered within the scope of the TCP/IP Interface Object behavior. Devices are free to implement such behavior, however it shall be considered to have occurred in the “Non-Existent” state.

Figure 5-3.1 Diagram Showing the Behavior of the TCP/IP Object





5-4 Ethernet Link Object

Class Code: F6 $_{hex}$

5-4.1 Scope

The Ethernet Link Object maintains link-specific counters and status information for an IEEE 802.3 communications interface. Each device shall support exactly one instance of the Ethernet Link Object for each IEEE 802.3 communications interface on the module. Devices may use an Ethernet Link Object instance for an internally accessible interface, such as an internal port for an embedded switch, Refer to Chapter 6 (Device Profiles) for additional information on multi-port EtherNet/IP devices.

5-4.2 Revision History

Since the initial release of this object class definition changes have been made that require a revision update of this object class. The table below represents the revision history.

Table 5-4.1 Revision History

Revision	Reason for Object Definition Update
1	Initial revision of this object definition
2	Add Instance Attribute 6, Interface Control
3	Add new instance attributes 7-10 providing support for multiple port Ethernet devices

5-4.3 Attributes

5-4.3.1 Class Attributes

The Ethernet Link Object shall support the following class attributes.

Table 5-4.2 Class Attributes

Attr ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
1	Conditional [1]	Get	NV	Revision	UINT	Revision of this object	The minimum value shall be 1. Shall be 2 or greater if instance attribute 6 is implemented. Shall be 3 if any instance attributes 7-10 are implemented. The maximum value shall be 3.
2	Conditional [2]	Get	NV	Max Instance	UINT	Maximum instance number of an object currently created in this class level of the device	The largest instance number of a created object at this class hierarchy level
3	Conditional [2]	Get	NV	Number of Instances	UINT	Number of object instances currently created at this class level of the device	The number of object instances at this class hierarchy level
4 thru 7	These class attributes are optional and are described in Volume 1, Chapter 4.						

1 Required if the Revision value is greater than 1.

2 Required if the number of instances is greater than 1.

An error reading the Class Revision attribute implies this is a revision 1 only implementation.

5-4.3.2 Instance Attributes

The Ethernet Link Object shall support the following instance attributes.

Table 5-4.3 Instance Attributes

Attr ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
1	Required	Get	V	Interface Speed	UDINT	Interface speed currently in use	Speed in Mbps (e.g., 0, 10, 100, 1000, etc.)
2	Required	Get	V	Interface Flags	DWORD	Interface status flags	Bit map of interface flags. See section 5-4.3.2.1
3	Required	Get	NV	Physical Address	ARRAY of 6 USINTs	MAC layer address	See section 5-4.3.2.3
4	Conditional[1]	Get	V	Interface Counters	STRUCT of:		See section 5-4.3.2.4
				In Octets	UDINT	Octets received on the interface	
				In Ucast Packets	UDINT	Unicast packets received on the interface	
				In NUcast Packets	UDINT	Non-unicast packets received on the interface	
				In Discards	UDINT	Inbound packets received on the interface but discarded	
				In Errors	UDINT	Inbound packets that contain errors (does not include In Discards)	
				In Unknown Protos	UDINT	Inbound packets with unknown protocol	
				Out Octets	UDINT	Octets sent on the interface	
				Out Ucast Packets	UDINT	Unicast packets sent on the interface	
				Out NUcast Packets	UDINT	Non-unicast packets sent on the interface	
				Out Discards	UDINT	Outbound packets discarded	
				Out Errors	UDINT	Outbound packets that contain errors	

Attr ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
5	Optional	Get	V	Media Counters	STRUCT of:	Media-specific counters	See section 5-4.3.2.5
				Alignment Errors	UDINT	Frames received that are not an integral number of octets in length	
				FCS Errors	UDINT	Frames received that do not pass the FCS check	
				Single Collisions	UDINT	Successfully transmitted frames which experienced exactly one collision	
				Multiple Collisions	UDINT	Successfully transmitted frames which experienced more than one collision	
				SQE Test Errors	UDINT	Number of times SQE test error message is generated	
				Deferred Transmissions	UDINT	Frames for which first transmission attempt is delayed because the medium is busy	
				Late Collisions	UDINT	Number of times a collision is detected later than 512 bit-times into the transmission of a packet	
				Excessive Collisions	UDINT	Frames for which transmission fails due to excessive collisions	
				MAC Transmit Errors	UDINT	Frames for which transmission fails due to an internal MAC sublayer transmit error	
				Carrier Sense Errors	UDINT	Times that the carrier sense condition was lost or never asserted when attempting to transmit a frame	
				Frame Too Long	UDINT	Frames received that exceed the maximum permitted frame size	
				MAC Receive Errors	UDINT	Frames for which reception on an interface fails due to an internal MAC sublayer receive error	
6	Optional	Set	NV	Interface Control	STRUCT of:	Configuration for physical interface	See section 5-4.3.2.6
				Control Bits	WORD	Interface Control Bits	
				Forced Interface Speed	UINT	Speed at which the interface shall be forced to operate	Speed in Mbps (10, 100, 1000, etc.)
7	Optional	Get	NV	Interface Type	USINT	Type of interface: twisted pair, fiber, internal, etc	See section 5-4.3.2.7

Attr ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
8	Optional	Get	V	Interface State	USINT	Current state of the interface: operational, disabled, etc	See section 5-4.3.2.8
9	Optional	Set	NV	Admin State	USINT	Administrative state: enable, disable	See section 5-4.3.2.9
10	Conditional [2]	Get	NV	Interface Label	SHORT_ STRING	Human readable identification	See section 5-4.3.2.10

1 The Interface Counters attribute is required if the Media Counters attribute is implemented.
2 Required if number of instances is greater than 1.

5-4.3.2.1 Interface Flags

The Interface Flags attribute contains status and configuration information about the physical interface and shall be as follows:

Table 5-4.4 Interface Flags

Bit(s):	Called:	Definition
0	Link Status	Indicates whether or not the IEEE 802.3 communications interface is connected to an active network. 0 indicates an inactive link; 1 indicates an active link. The determination of link status is implementation specific. In some cases devices can tell whether the link is active via hardware/driver support. In other cases, the device may only be able to tell whether the link is active by the presence of incoming packets.
1	Half/Full Duplex	Indicates the duplex mode currently in use. 0 indicates the interface is running half duplex; 1 indicates full duplex. Note that if the Link Status flag is 0, then the value of the Half/Full Duplex flag is indeterminate.
2-4	Negotiation Status	Indicates the status of link auto-negotiation 0 = Auto-negotiation in progress. 1 = Auto-negotiation and speed detection failed. Using default values for speed and duplex. Default values are product-dependent; recommended defaults are 10Mbps and half duplex. 2 = Auto negotiation failed but detected speed. Duplex was defaulted. Default value is product-dependent; recommended default is half duplex. 3 = Successfully negotiated speed and duplex. 4 = Auto-negotiation not attempted. Forced speed and duplex.
5	Manual Setting Requires Reset	0 indicates the interface can activate changes to link parameters (auto-negotiate, duplex mode, interface speed) automatically. 1 indicates the device requires a Reset service be issued to its Identity Object in order for the changes to take effect.
6	Local Hardware Fault	0 indicates the interface detects no local hardware fault; 1 indicates a local hardware fault is detected. The meaning of this is product-specific. Examples are an AUI/MII interface detects no transceiver attached or a radio modem detects no antennae attached. In contrast to the soft, possible self-correcting nature of the Link Status being inactive, this is assumed a hard-fault requiring user intervention.
7-31	Reserved	Shall be set to zero

5-4.3.2.2 Interface Speed

The Interface Speed attribute shall indicate the speed at which the interface is currently running (e.g., 10 Mbps, 100 Mbps, 1 Gbps, etc.) A value of 0 shall be used to indicate that the speed of the interface is indeterminate. The scale of the attribute is in Mbps, so if the interface is running at 100 Mbps then the value of Interface Speed attribute shall be 100. The Interface Speed is intended to represent the media bandwidth; the attribute shall not be doubled if the interface is running in full-duplex mode.

5-4.3.2.3 Physical Address

The Physical Address attribute contains the interface's MAC layer address. The Physical Address is an array of octets. The recommended display format is "XX-XX-XX-XX-XX-XX", starting with the first octet. Note that the Physical Address is not a settable attribute. The Ethernet address shall be assigned by the manufacturer, and shall be unique per IEEE 802.3 requirements. Devices with multiple ports but a single MAC interface (e.g., a device with a embedded switch technology) may use the same value for this attribute in each instance of the Ethernet Link Object. The general requirement is that the value of this attribute shall be the MAC address used for packets to and from the device's own MAC interface over this physical port.

5-4.3.2.4 Interface Counters

The Interface Counters attribute contains counters relevant to the receipt of packets on the interface. These counters shall be as defined in RFC 1213 "MIB-II Management Information Base". The Interface Counters are a conditional attribute; they shall be implemented if the Media Counters attribute is implemented. Multi-port devices with a single MAC interface (e.g., device with an embedded switch) shall maintain counter values in one of three ways:

1 In each instance, count the MAC frames sent/received by the device itself over the port represented by that instance (i.e., ea ch physical interface counts the MAC frames sent/received over that interface). This is the preferred solution.

2 Use counter values of 0 for those instances that correspond to the external switch ports; count MAC frames in the instance that corresponds to the internal device interface

3 Use the same counter values for all instances, counting MAC frames sent/received by the device itself

5-4.3.2.5 Media Counters

The Media Counters attribute contains counters specific to Ethernet media. These counters shall be as defined by RFC 1643, "Definitions of Managed Objects for Ethernet-Like Interface Types". If this attribute is implemented the Interface Counters shall also be implemented. Instances that refer to internal interfaces may set the values of the Interface Counters to 0.

Note: some underlying hardware or system implementations may not provide all of the Media Counters. In the case of fiber media, some of the counters do not apply (e.g., collision counters). Devices shall use values of 0 for counters that are not implemented.

5-4.3.2.6 Interface Control

The Interface Control attribute is a structure consisting of Control Bits and Forced Interface Speed and shall be as follows:

5-4.3.2.6.1 Control Bits

Table 5-4.5 Control Bits

Bit(s):	Called:	Definition
0	Auto-negotiate	0 indicates 802.3 link auto-negotiation is disabled. 1 indicates auto-negotiation is enabled. If auto-negotiation is disabled, then the device shall use the settings indicated by the Forced Duplex Mode and Forced Interface Speed bits.
1	Forced Duplex Mode	If the Auto-negotiate bit is 0, the Forced Duplex Mode bit indicates whether the interface shall operate in full or half duplex mode. 0 indicates the interface duplex should be half duplex. 1 indicates the interface duplex should be full duplex. Interfaces not supporting the requested duplex shall return a GRC hex 0x09 (Invalid Attribute Value). If auto-negotiation is enabled, attempting to set the Forced Duplex Mode bits shall result in a GRC hex 0x0C (Object State Conflict).
2-15	Reserved	Shall be set to zero

5-4.3.2.6.2 Forced Interface Speed

If the Auto-negotiate bit is 0, the Forced Interface Speed bits indicate the speed at which the interface shall operate. Speed is specified in megabits per second (e.g., for 10 Mbps Ethernet, the Interface Speed shall be 10). Interfaces not supporting the requested speed should return a GRC hex 0x09 (Invalid Attribute Value).

If auto-negotiation is enabled, attempting to set the Forced Interface Speed shall result in a GRC hex 0x0C (Object State Conflict).

5-4.3.2.7 Interface Type

The Interface Type attribute shall indicate the type of the physical interface. Table 5-4.6 shows the Interface Type values. This attribute shall be stored in non-volatile memory.

Table 5-4.6 Interface Type

Value	Type of interface
0	Unknown interface type.
1	The interface is internal to the device, for example, in the case of an embedded switch.
2	Twisted-pair (e.g., 10Base-T, 100Base-TX, 1000Base-T, etc.)
3	Optical fiber (e.g., 100Base-FX)
4-255	Reserved.

5-4.3.2.8 Interface State

The Interface State attribute shall indicate the current operational state of the interface. Table 5-4.7 shows the Interface State values. This attribute shall be stored in volatile memory.

Table 5-4.7 Interface State

Value	Interface State
0	Unknown interface state
1	The interface is enabled and is ready to send and receive data
2	The interface is disabled
3	The interface is testing
4-255	Reserved.

5-4.3.2.9 Admin State

The Admin State attribute shall allow administrative setting of the interface state. Table 5-4.8 shows the Admin State values. This attribute shall be stored in non-volatile memory.

Table 5-4.8 Admin State

Value	Admin State
0	Reserved
1	Enable the interface
2	Disable the interface. If this is the only CIP communications interface, a request to disable the interface shall result in an error (status code 0x09).
3-255	Reserved.

5-4.3.2.10 Interface Label

The Interface Label attribute shall be a text string that describes the interface. The content of the string is vendor specific. For internal interfaces the text string should include “internal” somewhere in the string. The maximum number of characters in this string is 64. This attribute shall be stored in non-volatile memory.

5-4.4 Common Services

5-4.4.1 All Services

The Ethernet Link Object shall provide the following common services.

Table 5-4.9 Common Services

Service	Need in Implementation			
Code	Class	Instance	Service Name	Description of Service
0x01	Optional	Optional	Get_Attributes_All	Returns a predefined listing of this objects attributes (See the Get_Attributes_All response definition in section 5-4.4.2)
0x0E	Conditional	Required	Get_Attribute_Single	Returns the contents of the specified attribute.
0x10	n/a	Conditional	Set_Attribute_Single	Modifies a single attribute.

The Get_Attribute_Single service shall be implemented for the class if any class attribute is implemented.

The Set_Attribute_Single service shall be implemented if the Interface Control or Admin State attributes are implemented.

5-4.4.2 Get_Attributes_All Response

At the class level, the Get_Attributes_All response shall contain the class attributes in numerical order, up to the last implemented attribute. Any unimplemented attributes in the response shall use the default attribute values.

For instance attributes, attributes shall be returned in numerical order, up to the last implemented attribute. The Get_Attributes_All reponse shall be as follows:

Table 5-4.10 Get_Attributes_All Response

Attribute ID	Size in Bytes	Contents
1	4	Interface Speed
2	4	Interface Flags
3	6	Physical Address
4	44	Interface Counters Default value of 0 if not implemented
5	48	Media Counters Default value of 0 if not implemented
6	4	Interface Control Default value of 0 if not implemented Note: A value of 0 (which would literally indicate Auto-negotiate disabled, half-duplex, but speed of 0) shall indicate that the attribute is not implemented.
7	1	Interface Type Default value of 0 if not implemented.
8	1	Interface State Default value of 0 if not implemented.
9	1	Admin State Default value of 0 if not implemented.
10	1	Interface Label Length
	Variable, equal to Interface Label Length	Interface Label

The length of the Interface Label is not known before issuing the Get_Attribute_All service request. Implementers shall be prepared to accept a response containing the maximum size of the Interface Label (65 USINTs).

5-4.5 Class-Specific Services

The Ethernet Link Object shall support the following class-specific services:

Table 5-4.11 Class Specific Services

Service Code	Need in Implementation		Service Name	Description of Service
	Class	Instance		
0x4C	n/a	Conditional [1]	Get_and_Clear	Gets then clears the specified attribute (Interface Counters or Media Counters).

1 The Get_and_Clear service shall only be implemented if the Interface Counters and Media Counters are implemented.

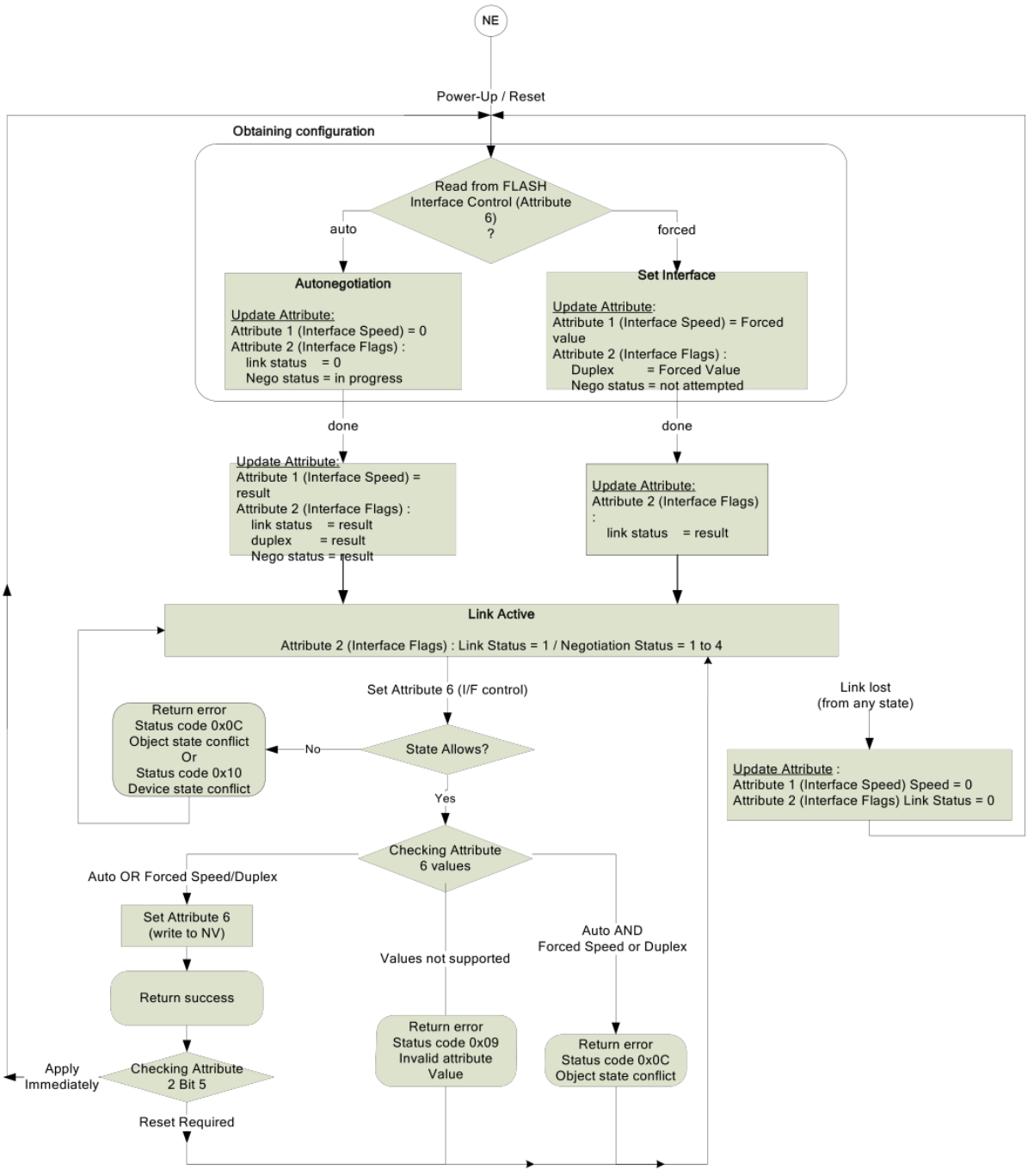
5-4.5.1 Get_and_Clear Service

The Get_and_Clear service is a class-specific service. It is only supported for the Interface Counters and Media Counters attributes. The Get_and_Clear response shall be the same as the Get_Attribute_Single response for the specified attribute. After the response is built, the value of the attribute shall be set to zero.

5-4.6 Behavior

The behavior of the Ethernet Link Object shall be as illustrated in Diagram below.

Figure 5-4.1 Diagram Showing the Behavior of the Ethernet Link Object



5-5 Device Level Ring (DLR) Object

Class Code: 47 hex

5-5.1 Scope

The Device Level Ring (DLR) Object provides the configuration and status information interface for the DLR protocol. The DLR protocol is a layer 2 protocol that enables the use of an Ethernet ring topology. The DLR protocol is fully specified in Chapter 9. The DLR Object provides the CIP application-level interface to the protocol.

The DLR Object shall be implemented in all multi-port EtherNet/IP devices that support the Device Level Ring protocol.

Devices shall implement no more than one instance of the DLR Object. Support for the DLR protocol on multiple pairs of ports is future enhancement that is at the present time undefined.

5-5.2 Revision History

Since the initial release of this object class definition changes have been made that require a revision update of this object class. The table below represents the revision history:

Revision	Description
1	Initial Definition at First Release of Specification (obsolete)
2	Instance Attribute 10 changed to Required (vs. Conditional) and added to non-supervisor Get_Attributes_All response. Added (Required) instance attribute 12. Added attribute 12 to Get_Attributes_All responses. Add Restart_Sign_On Conditional Service to Object Specific services.

5-5.3 Attributes

5-5.3.1 Class Atributes

Table 5-5.1 Class Attributes

Attrib ID	Need in Implementation	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
1	Conditional [1]	Get	NV	Revision	UINT	Revision of this object	The current value assigned to this attribute is 2
2 thru 7	These class attributes are optional and are described in Volume 1, Chapter 4 of the CIP Common specification.						

1 Required if the Revision value is greater than 1

5-5.3.2 Instance Attributes

Table 5-5.2 Instance Attributes

Attrib ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
1	Required	Get	V	Network Topology	USINT	Current network topology mode	0 indicates “Linear” 1 indicates “Ring”
2	Required	Get	V	Network Status	USINT	Current status of network	0 indicates “Normal” 1 indicates “Ring Fault” 2 indicates “Unexpected Loop Detected” 3 indicates “Partial Network Fault” 4 indicates “Rapid Fault/Restore Cycle”
3	Conditional [1]	Get	V	Ring Supervisor Status	USINT	Ring supervisor active status flag	0 – indicates the node is functioning as a backup 1 – indicates the device is functioning as the active ring supervisor. 2 – indicates the device is functioning as a normal ring node. 3 – indicates the device is operating in a non-DLR topology 4 – indicates the device cannot support the currently operating ring parameters (Beacon Interval and/or Beacon Timeout)

Device Level Ring (DLR) Object, Class Code: 47_{Hex}

Attrib ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
4	Conditional [1]	Set	NV	Ring Supervisor Config	Struct of:	Ring Supervisor configuration parameters	
				Ring Supervisor Enable	BOOL	Ring supervisor enable flag	TRUE indicates the device is configured as a ring supervisor. FALSE indicates device is configured as a normal ring node Default=FALSE
				Ring Supervisor Precedence	USINT	Precedence of a ring supervisor in network with multiple ring supervisors	Numerically higher value indicates higher precedence Default=0
				Beacon Interval	UDINT	Duration of ring beacon interval	Beacon interval in microseconds. Default=400 microseconds
				Beacon Timeout	UDINT	Duration of ring beacon timeout	Beacon timeout in microseconds. Default=1960 microseconds
				DLR VLAN ID	UINT	VLAN ID to use in ring protocol messages	Value range is 0-4094 Default=0
5	Conditional [1]	Set	V	Ring Faults Count	UINT	Number of ring faults since power up	
6	Conditional [1]	Get	V	Last Active Node on Port 1	STRUCT of:	Last active node at the end of chain through port 1 of active ring supervisor during ring fault	
					UDINT	Device IP Address	A value of 0 indicates no IP Address has been configured for the device. Initial value shall be 0.
					ARRAY of 6 USINTs	Device MAC Address	Ethernet MAC address
7	Conditional [1]	Get	V	Last Active Node on Port 2	STRUCT of:	Last active node at the end of chain through port 2 of active ring supervisor during ring fault	
					UDINT	Device IP Address	A value of 0 indicates no IP Address has been configured for the device
					ARRAY	Device MAC	Ethernet MAC address

Attrib ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of Values
					of 6 USINTs	Address	
8	Conditional [1]	Get	V	Ring Protocol Participants Count	UINT	Number of devices in ring protocol participants list	
9	Conditional [1]	Get	V	Ring Protocol Participants List	ARRAY of :	List of devices participating in ring protocol	
					STRUCT of:		
					UDINT	Device IP Address	A Value of 0 indicates no IP Address has been configured for the device
					ARRAY of 6 USINTs	Device MAC Address	Ethernet MAC address
10	Required	Get	V	Active Supervisor Address	STRUCT of:	IP and/or MAC address of the active ring supervisor	
					UDINT	Supervisor IP Address	A Value of 0 indicates no IP Address has been configured for the device
					ARRAY of 6 USINTs	Supervisor MAC Address	Ethernet MAC address
11	Conditional [1]	Get	V	Active Supervisor Precedence	USINT	Precedence value of the active ring supervisor	
12	Required	Get	NV	Capability Flags	DWORD	Describes the DLR capabilities of the device	See section 5-5.3.14

1 Shall be implemented for devices capable of functioning as a ring supervisor. Shall not be implemented by non-supervisor devices.

5-5.3.3 Network Topology

The Network Topology attribute indicates the current network topology mode. A value of 0 shall indicate “Linear” topology. A value of 1 shall indicate “Ring” topology. The value of the attribute shall correspond to the DLR State specified in Chapter 9.

When a supervisor-capable device is enabled as a ring supervisor, the Network Topology attribute shall always indicate “Ring” except in the case where the device cannot support the current operating ring parameters. When a supervisor-capable device is not enabled as a ring supervisor, or is enabled as a ring supervisor but can’t support the current operating ring parameters, or the device is not a supervisor-capable device, the device shall initially indicate “Linear”, then shall transition between “Ring” and “Linear” modes as specified in Chapter 9.

5-5.3.4 Network Status

The Network Status attribute provides current status of the network based the device's view of the network, as specified in the DLR behavior in Chapter 9. Table 5-5.3 shows the possible values:

Table 5-5.3 Network Status values

Network Status value	Description
0	Normal operation in both Ring and Linear Network Topology modes.
1	Ring Fault. A ring fault has been detected. Valid only when Network Topology is Ring.
2	Unexpected Loop Detected. A loop has been detected in the network. Valid only when the Network Topology is Linear.
3	Partial Network Fault. A network fault has been detected in one direction only. Valid only when Network Topology is Ring and the node is the active ring supervisor.
4	Rapid Fault/Restore Cycle. A series of rapid ring fault/restore cycles has been detected, per the criteria in Chapter 9. Similar to the Partial Network Fault status, the supervisor remains in a state with forwarding blocked on its ring ports. The condition must be cleared explicitly via the "Clear Rapid Faults" service.

5-5.3.5 Ring Supervisor Status

The Ring Supervisor Status attribute indicates the device's status as a ring supervisor. Table 5-5.4 shows the possible values:

Table 5-5.4 Ring Supervisor Status values

Supervisor Status value	Description
0	Indicates the device is functioning as a backup supervisor.
1	Indicates the device is functioning as the active ring supervisor.
2	Indicates the device is functioning as a normal ring node (supervisor not enabled).
3	Indicates the device is functioning in a non-DLR topology (supervisor not enabled, and no other supervisor is present).
4	Indicates the device cannot support the currently operating ring parameters (Beacon Interval and/or Beacon Timeout).

5-5.3.6 Ring Supervisor Config

The Ring Supervisor Config attribute contains the configuration parameters needed for ring operation: Supervisor Precedence, Beacon Interval, Beacon Timeout, VLAN ID, Supervisor Enable/Disable.

If the device is the active supervisor, changes to the attribute shall be applied immediately. When the Supervisor Precedence, Beacon Interval, Beacon Timeout or VLAN ID are changed on the active ring supervisor, the ring supervisor shall cease sending Beacon frames for two beacon timeout periods then shall resume sending Beacon frames using the new parameters.

Backup ring supervisors shall obtain values for the Beacon Interval, Beacon Timeout and VLAN ID from the Beacon frame sent by the active ring supervisor, and shall store those values in non-volatile storage. If the obtained values cannot be supported by the device (e.g., Beacon Interval too small), the device shall set the Ring Supervisor Status attribute as noted in the attribute description, and shall not take over as active supervisor after a ring reconfiguration.

When the Ring Supervisor Config attribute is modified on a backup supervisor, the behavior depends on the backup supervisor's new precedence value compared to the active supervisor's precedence value:

- New backup precedence value is greater than the current active ring supervisor's precedence or of equal precedence with numerically higher MAC address than active supervisor MAC address: backup shall immediately begin sending Beacon frames with the new parameters.
- New backup precedence value is less than the active supervisor's precedence or of equal precedence with numerically lower MAC address than active supervisor MAC address: modification to the Beacon Interval, Beacon Timeout, and VLAN ID shall be ignored.

Attempts to set invalid Ring Supervisor Config values shall result in error code 0x09 (Invalid attribute value) returned from the set service, regardless of whether the device is an active or backup supervisor.

5-5.3.6.1.1 Ring Supervisor Enable

The Ring Supervisor Enable item enables or disables the ring supervisor function in a supervisor-capable device. A value of TRUE enables the supervisor function. A value of FALSE disables the supervisor function. The default value is FALSE.

5-5.3.6.1.2 Ring Supervisor Precedence

The Ring Supervisor Precedence item contains the user-assigned precedence value given to the ring supervisor. When multiple ring supervisors are enabled, the precedence value allows the user to configure the order in which the configured supervisors select the active supervisor.

The ring supervisor precedence must be chosen from the range 0-255, with numerically higher values indicating higher precedence. The default value shall be 0.

When more than one supervisor is enabled the supervisor with highest precedence becomes active ring supervisor, in accordance with Chapter 9. If multiple supervisors have the same precedence, the supervisor with the numerically higher MAC address becomes the active supervisor

5-5.3.6.1.3 Beacon Interval

The Beacon Interval item contains the interval in microseconds that the ring supervisor shall use in generating beacon frames. Per the DLR protocol specification in Chapter 9, the default value shall be 400 microseconds. Supervisors shall support a range from 400 microseconds to 100 milliseconds. Supervisors may support a Beacon Interval smaller than 400 microseconds, but this is not required. The absolute minimum Beacon Interval is 100 microseconds.

5-5.3.6.1.4 Beacon Timeout

The Beacon Timeout item contains the number of microseconds the ring supervisor shall wait for a beacon frame before declaring a beacon timeout.

The default value shall be 1960 microseconds, which is based on a nominal network size of 50 nodes and 100Mbps, full-duplex operation (refer to the Performance Calculations in Chapter 9). The user may wish to change the Beacon Timeout for other exceptional network circumstances (e.g., very large networks or very small high-performance motion networks).

The Beacon Timeout shall be at least 2 times the Beacon Interval value. If the Beacon Interval is changed and the Beacon Timeout becomes less than 2 times the Beacon Interval, the supervisor shall adjust the Beacon Timeout to be 2 times the Beacon Interval.

Supervisors shall support a range from 800 microseconds to 500 milliseconds. Supervisors may support a Beacon Timeout of smaller than 800 microseconds but this is not required. The absolute minimum Beacon Timeout is 200 microseconds.

5-5.3.6.1.5 DLR VLAN ID

The DLR VLAN ID contains the VLAN ID to be used in the DLR protocol frames. The DLR VLAN ID shall be used for all DLR protocol frames originated by the device, when the device is operating as the active ring supervisor. Devices that are not the active ring supervisor shall use the VLAN ID obtained from the active supervisor's frames (refer to Chapter 9 for additional details).

The VLAN ID value shall be in the range 0-4094. The default value shall be 0 (indicating no VLAN).

5-5.3.7 Ring Faults Count

The Ring Faults Count attribute contains the number of times since power up that the device has detected a ring fault, as either active or backup supervisor. If the Ring Supervisor Enable is set to FALSE, the Ring Faults Count shall be set to 0. The Ring Faults Count rolls over to 0 after it reaches its maximum value.

The attribute may also be reset to 0 via the Set_Attribute_Single service. Values other than 0 shall result in an error response.

5-5.3.8 Last Active Node on Port 1

The Last Active Node on Port 1 attribute contains the IP address and/or Ethernet MAC address of the last node reachable through port 1 of an active ring supervisor. The value of the attribute is obtained via the Link_Status/Neighbor Status frames, as specified in Chapter 9.

On transition to FAULT_STATE, this attribute shall remain clear until the supervisor receives Link/Neighbor Status information.

On transition from FAULT_STATE to NORMAL_STATE, the value of the attribute shall be retained, to aid in diagnosing the previous ring fault.

The initial values of IP address and Ethernet MAC address shall be 0. When the device is not enabled as a ring supervisor, or is operating as the backup supervisor the IP address and Ethernet MAC address shall be 0.

5-5.3.9 Last Active Node on Port 2

The Last Active Node on Port 2 attribute contains the IP address and/or Ethernet MAC address of the last node reachable through port 2 of an active ring supervisor. The value of the attribute is obtained via the Link_Status/Neighbor Status frames, as specified in Chapter 9.

On transition to FAULT_STATE, this attribute shall remain clear until the supervisor receives Link/Neighbor Status information.

On transition from FAULT_STATE to NORMAL_STATE, the value of the attribute shall be retained, to aid in diagnosing the previous ring fault.

The initial values of IP address and Ethernet MAC address shall be 0. When the device is not enabled as a ring supervisor, or is operating as the backup supervisor the IP address and Ethernet MAC address shall be 0.

5-5.3.10 Ring Protocol Participants Count

This attribute contains the number of members in the Ring Protocol Participants List attribute. The count and the list are gathered by the active ring supervisor through Sign_On frame as specified in Chapter 9.

If the device is not the active supervisor, the attribute shall be set to 0.

5-5.3.11 Ring Protocol Participants List

The Ring Protocol Participants List attribute contains the list of ring nodes participating in ring protocol. The participants list is gathered by the active ring supervisor via the Sign_On frame (see Chapter 9).

Since the size of the Ring Protocol Participants List could be large, depending on the number of nodes participating in the ring, this attribute shall be accessible with the Get_Member service.

Clients may elect to use the Get_Attribute_Single service to read the Ring Protocol Participants List. If the participants list is too large, the DLR Object shall return error code 0x11 (Reply Data Too Large).

If the device is not active supervisor, status code 0x0C (Object State Conflict) shall be returned.

If the number of participants received as a result of the Sign_On process exceeds the capacity of the supervisor's Ring Protocol Participants List, the last entry in the list shall be all 0xff's (0xffffffffffffffffffff).

5-5.3.12 Active Supervisor Address

This attribute contains the IP address and/or Ethernet MAC address of the active ring supervisor. The initial values of IP address and Ethernet MAC address shall be 0, until the active ring supervisor is determined.

5-5.3.13 Active Supervisor Precedence

This attribute contains the precedence value of the active ring supervisor. The initial value shall be 0, until the active ring supervisor is determined.

5-5.3.14 Capability Flags

The Capability Flags describe the DLR capabilities of the device.

Table 5-5.5 Capability Flags

Bit(s):	Called	Definition
0	Announce-based Ring Node[1]	Set if device's ring node implementation is based on processing of Announce frames. See Volume 2, Chapter 9-5.4.3 for more information.
1	Beacon-based Ring Node[1]	Set if device's ring node implementation is based on processing of Beacon frames. See Volume 2, Chapter 9-5.4.2 for more information.
2-4	Reserved	Shall be set to zero.
5	Supervisor Capable	Set if device is capable of providing the supervisor function.
6-31	Reserved	Shall be set to zero.

1 Bits 0 and 1 are mutually exclusive. Exactly only one of these bits shall be set in the attribute value that a device reports.

5-5.4 Common Services

Table 5-5.6 DLR Object Common Services

Service Code (Hex)	Need in Implementation		Service Name	Description of Service
	Class	Instance		
0x01	Optional	Required	Get_Attributes_All	Returns multiple attributes in numerical order.
0x0E	Optional	Required	Get_Attribute_Single	Returns a single attribute
0x10	n/a	Conditional [1]	Set_Attribute_Single	Modifies a single attribute
0x18	n/a	Conditional [2]	Get_Member	Returns members of attribute

1 This service must be implemented for devices capable of functioning as a ring supervisor

2 Required for access to the Ring Protocol Participants List. The member services extended protocol for multiple sequential members shall be supported (see Volume 1, Appendix A).

5-5.5 Get_Attributes_All Response

5-5.5.1 Class Level

At the class level, the Get_Attributes_All response shall contain the class attributes in numerical order, up to the last implemented attribute. Any unimplemented attributes in the response shall use the default attribute values.

5-5.5.2 Instance Level

At the instance level the Get_Attributes_All response varies depending on whether or not the device is capable of being a ring supervisor. If the device is not capable of being a ring supervisor, the Get_Attributes_All response shall be as follows:

Table 5-5.7 Get_Attributes_All Response – non supervisor device

Attribute ID	Size in Bytes	Contents
1	1	Network Topology
2	1	Network Status
10	10	Active Supervisor Address
12	4	Capability Flags

Table 5-5.8 Get _Attributes_All Response – supervisor-capable device

Attribute ID	Size in Bytes	Contents
1	1	Network Topology
2	1	Network Status
3	1	Ring Supervisor Status
4	12	Ring Supervisor Config:
5	2	Ring Faults Count
6	10	Last Active Node on Port 1
7	10	Last Active Node on Port 2
8	2	Ring Protocol Participants Count
10	10	Active Supervisor Address
11	1	Active Supervisor Precedence
12	4	Capability Flags

Note: Attribute 9 (Ring Protocol Participants List) is not included in the Get-Attributes-All response due to its potential size.

5-5.6 Class-Specific Services

The DLR Object shall support the following class-specific services:

Table 5-5.9 DLR Object Class-Specific Services

Service Code (Hex)	Need in Implementation		Service Name	Description of Service
	Class	Instance		
0x4B	n/a	Conditional [1]	Verify_Fault_Location	Causes ring supervisor to verify fault location by issuing Locate_Fault ring protocol message to ring nodes and update Last Active Node 1 and Last Active Node 2 attributes.
0x4C	n/a	Conditional [1]	Clear_Rapid_Faults	Clears the Rapid Fault/Restore Cycle Detected condition in the ring supervisor, allowing the supervisor to return to normal operation.
0x4D	n/a	Conditional [1]	Restart_Sign_On	Restart Sign On process and refresh DLR participants list.

1 This service shall be implemented for devices capable of functioning as a ring supervisor

5-5.6.1 Verify_Fault_Location Service

The Verify_Fault_Location service shall cause an active ring supervisor to verify a ring fault location by retransmitting the Locate_Fault frame to ring nodes (see Chapter 9). The Last Active Node 1 and Last Active Node 2 attributes shall be updated based on the response to the Locate_Fault frame.

There are no parameters for either the Verify_Fault_Location request or reply.

If the Verify _Fault_Location service is received when the supervisor is not enabled, or is currently the backup supervisor, or is the active supervisor but not in fault state, status code 0x0C (Object State Conflict) shall be returned, and the Last Active Node 1 and Last Active Node 2 attributes shall be set to 0.

5-5.6.2 Clear_Rapid_Faults Service

The Clear_Rapid_Faults service shall clear the condition where the ring supervisor has detected a cycle of rapid ring fault/restore (as defined in Chapter 9). Upon clearing the condition, the ring supervisor shall return to normal operation.

If the Clear_Rapid_Faults service is received when the supervisor is not enabled, or is currently the backup supervisor, or is the active supervisor but not in the rapid fault/restore condition, status code 0x0C (Object State Conflict) shall be returned.

5-5.6.3 Restart_Sign_On Service

The Restart_Sign_On service shall restart Sign On process (see Volume 2, Chapter 9-5.5.2.3 Sign On for more information) by active ring supervisor, if it is not currently in progress.

If the Restart_Sign_On service is received when the supervisor is not enabled, or is currently the backup supervisor, or is the active supervisor but not in the NORMAL_STATE, status code 0x0C (Object State Conflict) shall be returned. If the Restart_Sign_On service is received when a prior Sign On process is in progress, success shall be returned and the request shall be ignored

5-6 QoS Object

Class Code: 48 hex

5-6.1 Overview

Quality of Service (QoS) is a general term that is applied to mechanisms used to treat traffic streams with different relative priorities or other delivery characteristics. Standard QoS mechanisms include IEEE 802.1D/Q (Ethernet frame priority) and Differentiated Services (DiffServ) in the TCP/IP protocol suite.

The QoS Object provides a means to configure certain QoS-related behaviors in EtherNet/IP devices.

The QoS Object is required for devices that support sending EtherNet/IP messages with non-zero DiffServ code points (DSCP), or sending EtherNet/IP messages in 802.1Q tagged frames.

Refer to Volume 2, Chapter 3 for EtherNet/IP device behavior related to QoS.

5-6.2 Revision History

Table 5-6.1 Revision History

Revision	Hstory
01	Initial Definition

5-6.3 Class Attributes

The QoS object shall support the following class attributes.

Table 5-6.2 Class Attributes

Attribute ID	Need in Implementation	Access Rule	Name	Data Type	Description of Attribute	Semantics of values
1 thru 7	These class attributes are optional and are described in Chapter 4 of Volume 1 (the CIP Common specification).					

5-6.4 Instance Attributes

The QoS object shall support the following instance attributes.

Table 5-6.3 QoS Object Instance Attributes

Attr ID	Need in Implem	Access Rule	NV	Name	Data Type	Description of Attribute	Semantics of values
1	Conditional [1]	Set	NV	802.1Q Tag Enable	USINT	Enables or disables sending 802.1Q frames on CIP and IEEE 1588 messages	A value of 0 indicates tagged frames disabled. A value of 1 indicates tagged frames enabled. The default value shall be 0.
2	Conditional [2]	Set	NV	DSCP PTP Event	USINT	DSCP value for PTP (IEEE 1588) event messages	Refer to section 5-6.4.2
3	Conditional [2]	Set	NV	DSCP PTP General	USINT	DSCP value for PTP (IEEE 1588) general messages	Refer to section 5-6.4.2
4	Required	Set	NV	DSCP Urgent	USINT	DSCP value for CIP transport class 0/1 Urgent priority messages	Refer to section 5-6.4.2
5	Required	Set	NV	DSCP Scheduled	USINT	DSCP value for CIP transport class 0/1 Scheduled priority messages	Refer to section 5-6.4.2
6	Required	Set	NV	DSCP High	USINT	DSCP value for CIP transport class 0/1 High priority messages	Refer to section 5-6.4.2
7	Required	Set	NV	DSCP Low	USINT	DSCP value for CIP transport class 0/1 low priority messages	Refer to section 5-6.4.2
8	Required	Set	NV	DSCP Explicit	USINT	DSCP value for CIP explicit messages (transport class 2/3 and UCMM)	Refer to section 5-6.4.2

1 Required if the device supports sending 802.1Q frames
2 Required if the device supports CIP Sync

5-6.4.1 802.1Q Tag Enable

The 802.1Q Tag Enable attribute enables or disables sending 802.1Q frames on CIP and IEEE 1588 messages. When the attribute is enabled, the device shall send 802.1Q frames for all CIP and IEEE 1588 messages. The 802.1Q priority value shall be as specified in Volume 2, Chapter 3.

A value of 1 shall indicate enabled. A value of 0 shall indicate disabled. The default value for the attribute shall be 0. A change to the value of the attribute shall take effect the next time the device restarts.

Note: devices shall always use the corresponding DSCP values regardless of whether 802.1Q frames are enabled or disabled.

5-6.4.2 DSCP Value Attributes

Attributes 2 through 8 contain the DSCP values that shall be used for the different types of EtherNet/IP traffic.

Volume 2, Chapter 3-7 shows the format of the DSCP value within the IP header. Since the DSCP field has a size of 6 bits, the valid range of values for these attributes is 0-63. Note that the DSCP value, if placed directly in the ToS field in the IP header, must be shifted left 2 bits.

Table 5-6.4 shows the default DSCP values and traffic usages.

Table 5-6.4 Default DCSP Values and Usages

Attribute	Traffic Type Usage	Default DSCP
DSCP PTP Event	PTP (IEEE 1588) event messages	59 (‘111011’)
DSCP PTP General	PTP (IEEE 1588) general messages	47 (‘101111’)
DSCP Urgent	CIP transport class 0/1 messages with Urgent priority	55 (‘110111’)
DSCP Scheduled	CIP transport class 0/1 messages with Scheduled priority	47 (‘101111’)
DSCP High	CIP transport class 0/1 messages with High priority	43 (‘101011’)
DSCP Low	CIP transport class 0/1 messages with Low priority	31 (‘011111’)
DSCP Explicit	CIP UCMM CIP class 3	27 (‘011011’)

A change to the value of the above attributes shall take effect the next time the device restarts.

5-6.5 Common Services

The QoS Object provides the following common services.

Table 5-6.5 Common Services

Service	Need in Implementation		Service	Description of
Code	Class	Instance	Name	Service
0x01	Optional	N/A	Get_Attributes_All	See Volume 1, Appendix A
0x0E	Conditional [1]	Required	Get_Attribute_Single	See Volume 1, Appendix A
0x10	N/A	Required	Set_Attribute_Single	See Volume 1, Appendix A

1 Required if any class attributes are implemented

5-6.6 Get_Attributes_All Response

5-6.6.1 Class Level

The Get_Attributes_All response shall contain the class attributes in numerical order, up to the last implemented attribute. Any unimplemented attributes in the response shall use the default attribute values.

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 6: Device Profiles

Contents

6-1 Introduction

This chapter of the EtherNet/IP specification contains device profiles that are EtherNet/IP specific.

6-2 Required Objects

At minimum, every EtherNet/IP device shall implement instance number one of each of the following objects:

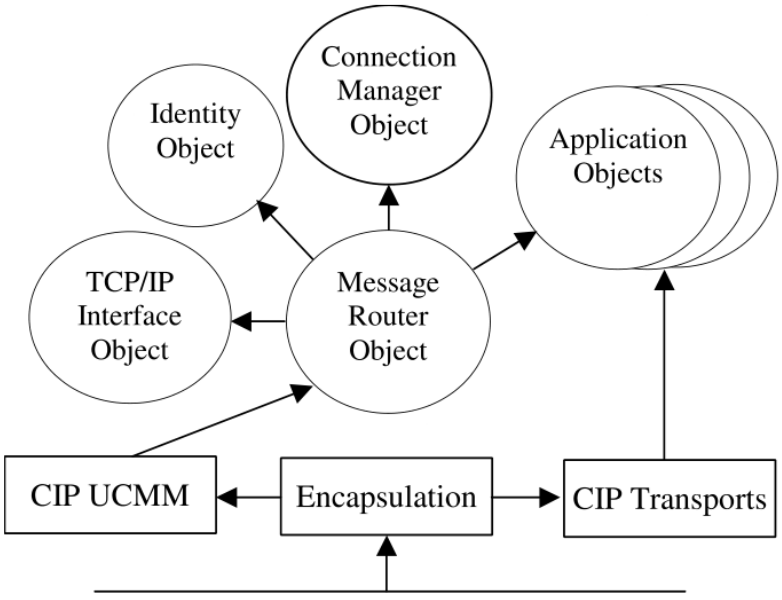
- Identity Object (class code = 0x01)
- Message Router Object (class code = 0x02)
- Connection Manager Object (class code = 0x06)
- TCP/IP Interface Object (class code = 0xF5) and a corresponding link object

If an Ethernet medium is used, the corresponding link object shall be the Ethernet Link Object (class code = 0xF6). If any other medium is used, the vendor shall define a vendor specific link object.

NOTE: This specification permits the use of any medium that supports TCP/IP; however, only the Ethernet medium has been completely standardized here. It is likely in the future that ODVA/CI will standardize other link objects for frequently used TCP/IP media. For example, in the future, a standardized PPP object may be defined.

Although it does not have an object class code, each device shall also implement the CIP Unconnected Message Manager (UCMM).

Figure 6-2.1 Base Device Object Model



6-3 Devices with Multiple Interfaces

EtherNet/IP devices may implement multiple network interfaces, for example:

- A device with a single IP address with two Ethernet ports implemented as an embedded switch
- An EtherNet/IP-enabled switch with multiple Ethernet ports and with EtherNet/IP communications for the switch itself
- A device with a single Ethernet interface and multiple IP addresses

Note: The specification does not address any behavior related to the Ethernet switching function in the device examples mentioned above. The intent of the specification at present is only to specify to allowable configurations of TCP/IP Interface Object and Ethernet Link Object instances in order to support the device possibilities above.

Devices with multiple interfaces shall implement multiple instances of the TCP/IP interface Object and physical link objects (e.g., Ethernet Link Object) as applicable to the device function as listed below:

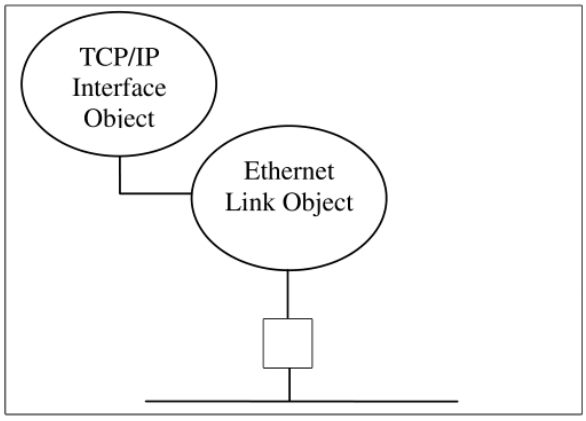
- 1 instance of the TCP/IP Interface Object for each TCP/IP interface (i.e., for each IP address).
- 1 instance of a physical link object (e.g., Ethernet Link Object) for each physical interface exposed via EtherNet/IP. Devices may elect not to expose all interfaces,
- Devices may use a physical link object instance (e.g., Ethernet Link Object) for an internal interface such as the internal device port of an embedded switch.
- Devices with multiple IP addresses may elect to represent each IP interface as a different CIP port, and allow CIP messages to be routed from one port to the other. For example, the CIP connection path would enter one of the ports and exit the other port. In this scenario, the device shall implement one instance of the Port Object for each CIP port, and shall implement the CIP routing mechanism described in Volume 1. Devices are not required to implement the Port Object unless they implement multiple CIP ports.

The following sections illustrate some of the different possibilities for devices with varying numbers of Ethernet interfaces.

6-3.1 Case 1: Single Port Device, 1 IP Address

This is the normal case for most EtherNet/IP devices : single Ethernet interface, single IP address. In this case, there is one instance of the TCP/IP Interface Object, and one instance of the Ethernet Link Object that represents the physical interface.

Figure 6-3.1 Case 1 Illustration

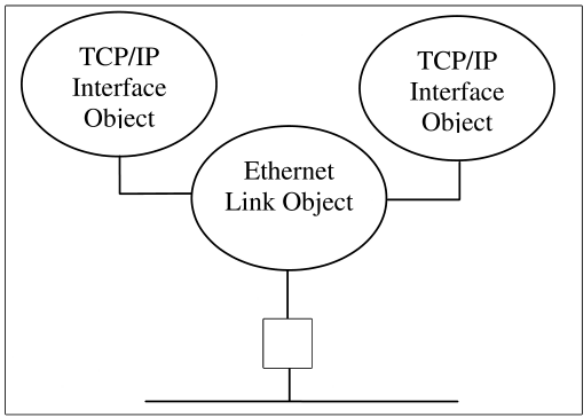


6-3.2 Case 2, Single Port Device, Multiple IP Addresses

Example : A device with a single Ethernet interface wishes to expose a second IP address.

In this example, there are 2 instances of the TCP/IP Interface Object, and one instance of the Ethernet Link Object that represents the physical interface. TCP/IP Interface Object class attributes 2 (Max. instances) and 3 (Number of instances) are used. The instance attribute 4 (Physical Link Object) of both instances refers to the same Ethernet Link Object

Figure 6-3.2 Case 2 Illustration

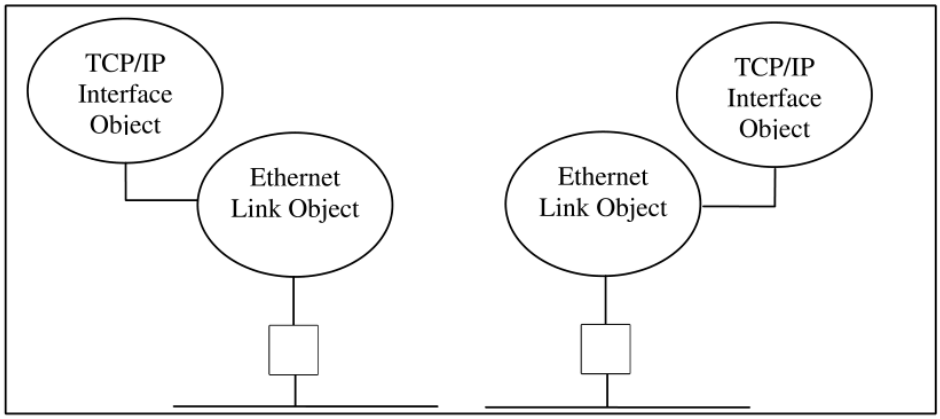


6-3.3 Case 3, Device with 2 Ethernet ports. Each port has an associated IP Address

Example : A device with a multiple Ethernet interfaces, each with an associated IP address. Each interface would be a different CIP-addressable port (i.e., there would be a Port Object instance per interface).

In this example, there are 2 instances of the TCP/IP Interface Object, and 2 instances of the Ethernet Link Object that represent each the physical interface. TCP/IP Interface Object / Ethernet Link Object class attributes 2 (Max. instances) and 3 (Number of instances) are used. The new Ethernet Link instance attribute 10 (Interface Label) is used to get the correlation between the Ethernet Link Object and the physical port.

Figure 6-3.3 Case 3 Illustration



Note: this example can be generalized to more than 2 ports.

6-3.4 Case 4, Device with Multiple Ethernet interfaces and a single IP Address and CIP Interface

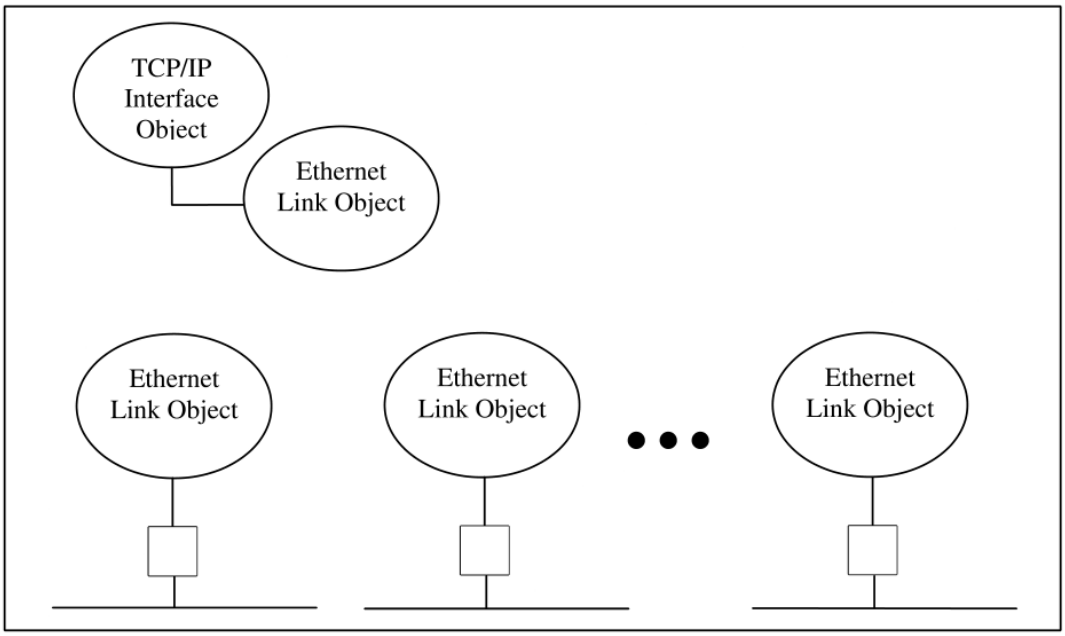
Example:An example is a device with embedded switch technology (to support linear topology), or an EtherNet/IP-enabled switch. In this case, the device has multiple Ethernet interfaces, but the interfaces are not CIP-addressable ports (i.e., they do not have corresponding CIP port numbers or Port Object instances).

It is however useful to allow configuration of the Ethernet interfaces, for example to set port speed and duplex via the Ethernet Link Object. Note that there is no intent to specify switching behavior of the device.

In this case, there is a single instance of the TCP/IP Interface Object, an optional “internal” instance of the Ethernet Link Object (corresponding to the internal device port), and then Ethernet Link Object instances for each of the physical interfaces.

New Ethernet Link Object class attributes 2 (Max. instances) and 3 (Number of instances) are used. The new Ethernet Link instance attribute 10 (Interface Label) is used to get the correlation between the Ethernet Link Object and the physical port. The new Ethernet Link instance attribute 7 (Interface Type) defines the kind of object (internal/external).

Figure 6-3.4 Case 4 Illustration



Refer to Chapter 5 (Object Library) for specific EtherNet/IP object definitions and requirements.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 7: Electronic Data Sheets

Contents

7-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the definition of electronic data sheets (EDS) that are EtherNet/IP specific. See the CIP Common specification for more information about the format of electronic data sheets and the definition of EDS related terms such as EDS section, EDS entry and EDS field.

7-2 Common Object Class Information

7-2.1 Object Class Sections

The following table identifies section keywords that are used to publicize EtherNet/IP object class information. Each of the following EDS Section Keywords may contain any of the Common Object Class Entry Keyword(s) described in Volume 1, section 7-3.6.1.2. The following table contains EtherNet/IP Object Class Section Names.

Table 7-2.1 Object Class Section Keywords

CIP Object	Section Keyword Name	Required/Optional	See section
TCP/IP Interface Object F5 hex	[TCP/IP Interface Class]	Conditional [1]	7-6
EtherNet Link Object F6 hex	[Ethernet Link Class]	Conditional [2]	7-2.2

1 Required if device supports Quick Connect functionality

2 Required if more than one Ethernet port exists on this device

7-2.2 [Ethernet Link Object Class] Section

The Ethernet Link Class section defines the characteristics of the Ethernet Link Object (see Object Library, Chapter 5) implemented in this device. The Common Object Class keywords may be used in the Ethernet Link Class section, see Volume 1, Chapter 7-3.6.1. The table below shows the additional entries in the Ethernet Link Class section. The Ethernet Link Class section begins with the keyword [Ethernet Link Class].

Table 7-2.2 Entry Keywords in the Ethernet Link Class Section

Entry Name	Entry Keyword	Required/Optional
Interface Label	InterfaceLabelN	Conditional [1]
Interface Type	InterfaceTypeN	Optional

1 Required if more than one Ethernet port exists on this device

7-2.2.1 Interface Label

The Interface label shall specify the Ethernet port interface label. The entry keyword for all Ethernet port interface labels shall consist of the character array “InterfaceLabel”, combined with a decimal number corresponding to an instance of the Ethernet link object. For example, InterfaceLabel1 is the Interface Label attribute of instance 1 of the Ethernet link object.

Table 7-2.3 Interface Label Keyword Format

Field Name	Field Number	Data Type	Required/Optional
Interface Label	1	STRING	Required

- Interface Label – specifies the name of this Ethernet port. The value of the Interface Label field shall be the same as the Interface Label attribute.

Example:

```
InterfaceLabel1 = "Port 1";
InterfaceLabel2 = "Port 2";
```

7-2.2.2 Interface Type

The Interface type shall specify the Ethernet port interface type. The entry keyword for all Ethernet port interface types shall consist of the character array "InterfaceType", combined with a decimal number corresponding to an instance of the Ethernet link object. For example, InterfaceType1 is the Interface Type attribute of instance 1 of the Ethernet link object.

Table 7-2.4 InterfaceType Keyword Format

Field Name	Field Number	Data Type	Required/Optional
Interface Type	1	USINT	Required

- Interface Type – specifies the type of this Ethernet port. See the Ethernet Link Object definition in Volume 2, chapter 5, "Interface Type" section, for a definition of the valid values for the Interface Type field. The value of the Interface Type field shall be the same as the Interface Type attribute.

Example:

```
InterfaceType1 = 2;      $ Twisted pair
InterfaceType2 = 3;      $ Optical fiber
```

7-3 [Device Classification] Section

In the [Device Classification] section of the EDS, for any EtherNet/IP compliant device, there shall be at least one ClassN keyword entry with its first field set to EthernetIP. As shown in Figure 7-4.1, no sub-classifications shall be present.

7-4 [Port] Section

In the [Port] section of the EDS (see Figure 7-4.1 for an example), the PortN entry corresponding to the EtherNet/IP compliant port shall be set as follows:

The "Port Type" field shall have a value of "TCP".

The optional "Port Object" field shall be set to the path of the TCP Object for this port.

No additional requirements, beyond those in the CIP Common Specification (Volume 1), are placed on the "Name" and "Port Number" fields.

NOTE: An EDS for an EtherNet/IP device does not directly refer to the link object for the EtherNet/IP port (for example, the Ethernet Link Object) since it can be referenced through the TCP Object for the port.

Figure 7-4.1 Example EDS of an EtherNet/IP Device

```
[File]
        DescText = "Widget EDS File";
        CreateData = 02-07-2001;
        CreateTime = 17:51:44;
        ModDate = 04-06-1997;
        ModTime = 22:07:30;
        Revision = 2.1;
        HomeURL = "http://www.odva.org/EDS/12345.eds";
[Device]
        VendCode = 65535;
        VendName = "Widget-Works, Inc.";
        ProdType = 0;
        ProdTypeStr = "Generic";
        ProdCode = 10;
        MajRev = 1;
        MinRev = 1;
        ProdName = "Smart-Widget";
        Catalog = "1492-SW";
        Icon = "widget.ico";

[Device Classification]
        Class1 = EthernetIP;

[Port]
        Port1 =
                TCP,
                "EtherNet/IP port",
                "20 F5 24 01",
                1;
```

7-5 DLR Class Section

The DLR Class section defines the characteristics of the DLR Object (see Object Library, Chapter 5) implemented in this device, if a DLR Object implementation exists. The Common Object Class keywords may be used in the DLR Class section (see chapter 7-3.6.1 Volume 1). The DLR Class section begins with the keyword [DLR Class]. This section is required if the device supports the DLR object.

Table 7-5.1 Additional Entry Keywords in the DLR Section

Entry Name	Entry Keyword	Required/Optional
Ring Supervisor Capable	Ring_Supervisor_Capable	Conditional[1]

1 Required if the device is capable of being the ring supervisor

7-5.1 Ring Supervisor Capable Entry keyword

The "**Ring_Supervisor_Capable**" entry keyword is conditional and shall contain the formatted field shown in Table 7-5.2. If the device is capable of being a ring supervisor the keyword is required. If the device is not capable of being a ring supervisor the keyword is optional. If this keyword is omitted, the device is not capable of being the ring supervisor.

Table 7-5.2 Ring_Supervisor_Capable Keyword Format

Field name	Field Number	Data type	Required/Optional
Supported	1	Field Keyword, possible values: Yes, No	Required

7-5.1.1 Supported, Field 1

This field indicates if the device can be the ring supervisor. If the field value is “Yes”, the device can be the ring supervisor. If the field value is “No”, the device can not be the ring supervisor.

7-6 TCP/IP Interface Class Section

The TCP/IP Interface Class section defines the characteristics of the TCP/IP Interface Object (see Volume 2, Object Library, Chapter 5) implemented in this device. The Common Object Class keywords may be used in the TCP/IP Interface Class section, see Volume 1, Chapter 7-3.6.1. The table below shows the additional conditional entries in the TCP/IP Interface Class section. The TCP/IP Interface Class section begins with the keyword [TCP/IP Interface Class].

Table 7-6.1 Entry Keywords in the TCP/IP Interface Class Section

Entry Name	Entry Keyword	Field Name	Required/Optional
EtherNet/IP Quick Connect Target	ENetQCTN	Ready for Connection Time	Conditional [1]
EtherNet/IP Quick Connect Originator	ENetQCON	Connection Origination Time	Conditional [2]

1 Required if a target device supports Quick Connect functionality

2 Required if a connection originator establishes connections to a Quick Connect target device

The entry keywords for all ports shall consist of the character array “ENetQCT” or “ENetQCO”, combined with a decimal number corresponding to an instance of the TCP/IP object. For example, ENetQCT1 is instance 1 of the TCP/IP Object.

7-6.1 ENetQCTN Keyword

Table 7-6.2 ENetQCTN Keyword Format

Field Name	Field Number	Data Type	Required/Optional
Ready for Connection Time	1	UINT	Required
CIP Connection Time	2	UINT	Required

The first field, "Ready for Connection Time", indicates the time, in milliseconds, from the application of device power to the production of the first Gratuitous ARP message, and its readiness to accept a TCP connection when the Quick Connect ports are set for forced speed and duplex and Auto-MDIX is disabled.

The second field, "CIP Connection Time" indicates the accumulated time in milliseconds for the following events:

a) Time from reception of ARP Request to ARP response

b) Time from reception of TCP open request to establishment of TCP connection

c) Time from reception of Register Session request to Register Session response

d) Time from reception of the Forward Open for the CIP I/O Connection to when it can produce its first I/O data message (this includes processing the Forward Open, applying any configuration data from the data segment, and generating the Forward Open response).

Example

```
TCP/IP Interface Class]
  ENetQCT1=350, 50;        $ 350ms Ready for Connection time
                           $ 50ms Accumulated CIP Connection Time
```

7-6.2 ENetQCON Keyword

Table 7-6.3 ENetQCON Keyword Format

Field Name	Field Number	Data Type	Required/Optional
Connection OriginationTime	1	UINT	Required

The field, "Connection Origination Time", indicates the accumulated time in milliseconds for the following events:

a) Time from when the controller receives the "Electrically Locked" input signal to when it can send the TCP open request (or ARP if one is needed).

b) Time from the establishment of the TCP connection to the Register Session request.

c) Time from reception of Register Session response to the Forward Open request.

d) Time from reception of the Forward Open response to the first output data on the I/O connection.

The connection origination time should be based on a best-case system where the system has to open up to seven connections to Quick Connect nodes on the tool. It is understood that if there are fewer or more than this number of nodes on the tool, this time will vary. The vendor of the controller shall describe in the product's documentation the conditions under which the best-case time was achieved and provide guidance as to the impact on this time under other conditions, e.g. - number of standard I/O connections, user program scan time, I/O connection performance (RPI) that provides the electrical-lock signal to the controller.

Example

```
[TCP/IP Interface Class]
  ENetQCO1=50;              $ 50ms Accumulated Connection Origination Time
```

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 8: Physical Layer

Contents

8-1 Introduction

Chapter 8 specifies EtherNet/IP media and physical layer requirements for EtherNet/IP installations and active devices. In some cases, industrial environmental requirements may exceed those used in office environments. Products and components may need to be enhanced to provide the level of performance required to support industrial applications. Some of these enhancements include noise rejection, sealing, voltage isolation, chemical resistance, shock, vibration and wide/dynamic temperature ranges.

8-2 General

The following sections will delineate physical layer media variants for EtherNet/IP. This standard does not define requirements for coaxial Ethernet components or commercial off the shelf components (COMMERCIAL). Requirements for these components can be found in ANSI IEEE 802.3 standard and TIA 568 standards. In this chapter, components that fall under these standards will be referred to as “standard components”. Systems constructed of standard components have been deployed in industrial environments primarily in information systems and limited control applications. These systems, for the most part, have been successfully providing services at 10 Mbps. Whether providing services at 10 Mbps or 100 Mbps, standard components are recognized and acceptable for use within the guidelines of this specification. However, because testing has shown that in order to survive harsh environments both in high noise, diverse temperatures and the presence of chemicals both in liquid and solid forms, there are system and component enhancements that are required.

This document defines component performance up to 100 Mbps. The component specifications herein are optimized for data rates of 10 Mbps and 100 Mbps. The copper variant shall include both shielded and unshielded twisted pair cable technologies. The signaling and coupling for copper twisted pair methods are described in section 8-9.2.1. Products constructed with Commercial components are not eligible for the industrial conformance check mark since the Commercial products cannot be directly mapped into any controlling standards controlled by ODVA. Products constructed of Commercial components are eligible for the Commercial checkmark.

8-3 Grounding (earthing) and Bonding

Grounding and bonding in the communications coverage area is critical to the performance of EtherNet/IP networks. This topic is a subject of further study.

8-4 Environmental Compatibility

Products and components installed in EtherNet/IP networks shall be compatible with the local environmental conditions ether by design or a combination of design and mitigation.

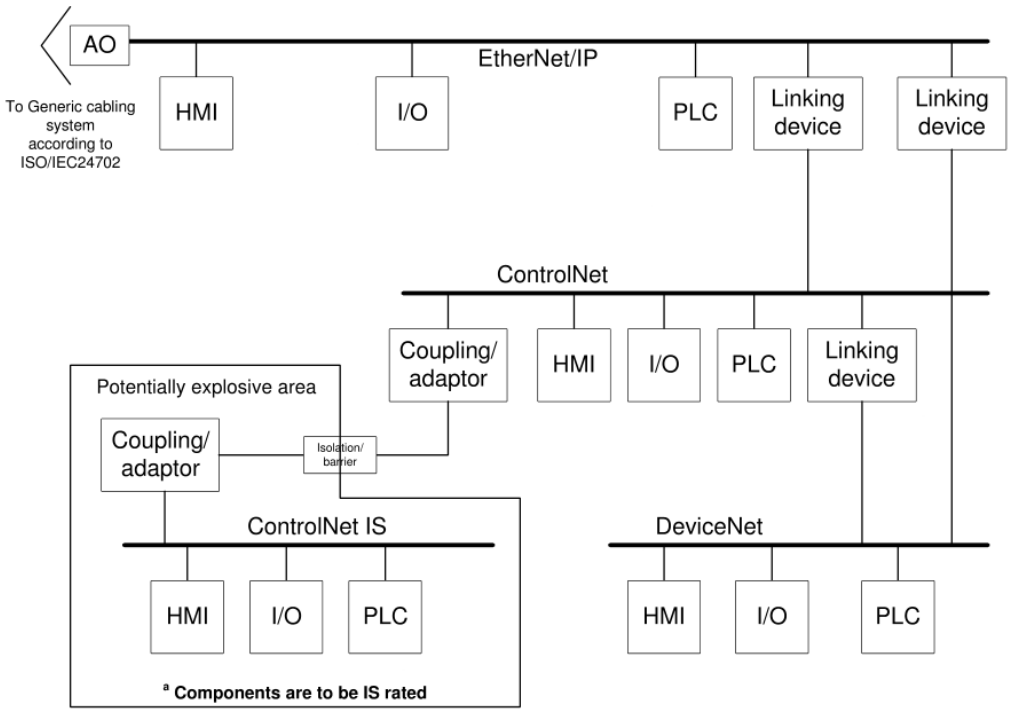
8-5 Auxiliary Power

For auxiliary power connector styles and pin outs see Section 8-2 and 8-3 of Volume 1 of the CIP Networks Library.

8-6 Supported Physical Topologies and relation to other networks

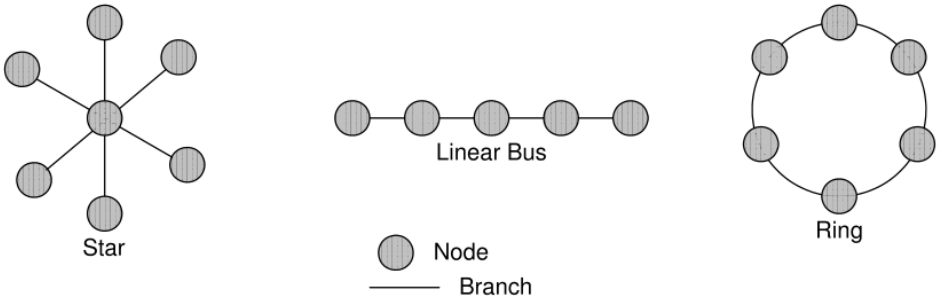
Figure 8-6.1 shows how EtherNet/IP maybe connected to other CIP networks and the generic telecommunications infrastructure. Connection to the generic telecommunications infrastructure should be through an appropriate security device to prevent inadvertent interruption to the control networks. The industrial generic cabling systems may not provide a level of performance required for control of industrial machinery and processes. The cabling is connected to the generic cabling as defined by ISO/IEC 24702 and TIA 1005 through an Automation Outlet (AO). The requirements of the automation outlet are defined in this chapter.

Figure 8-6.1 Relationship to Other Networks



There are three basic topologies for Ethernet based networks. EtherNet/IP supports all three of the standard active physical topologies as detailed in Figure 8-6.2.

Figure 8-6.2 Active Physical Topologies



A switching device is expected to be a dedicated device usually located in the center of the star physical topology. For other topologies such as the physical linear bus and ring the switching device may be embedded in a node. A physical linear bus and ring device requires two physical connections to the cabling infrastructure.

8-7 Performance Levels

Sections 8-7.1and 8-7.2 define two levels of product performance: Commercial EtherNet/IP and Industrial EtherNet/IP enhanced components. COMMERCIAL cabling components may require mitigation in the form of isolation and separation from the various harsh elements found in a typical industrial environment. The EtherNet/IP industrial connectivity component requirements have added enhancements to reduce the level of mitigation that might otherwise be required. This clause also details the requirements for active devices both based on COMMERCIAL and industrially hardened in the same way as the cabling components.

NOTE: There are many sections within this chapter that specify optional requirements. This section distils these requirements into two distinct levels: commercial copper and fiber and industrial EtherNet/IP copper and fiber.

8-7.1 COMMERCIAL based EtherNet/IP products

8-7.1.1 Copper and Fiber Cabling Components

The requirements for COMMERCIAL cabling components are defined ANSI/TIA/EIA-568-B series standards and section 8-8. The use of COMMERCIAL components may degrade system performance. Use of such products or components may result in unsatisfactory performance in industrial control applications.

8-7.1.2 Active Interfaces (PMD)

Copper and fiber based COMMERCIAL active products shall meet the minimum requirements of this chapter. Since these devices are not expected to be industrial hardened, they may require additional mitigation when installed in a harsh environment. The copper interfaces shall provide a non-sealed RJ-45 jack at the network interface. The fiber interface shall provide connectivity to one of the non-sealed connectors detailed in 8-9.5.2.1 (LC, SC or ST).

8-7.2 Industrial EtherNet/IP products

8-7.2.1 EtherNet/IP Copper and Fiber Cabling Components

These components are designed to better withstand high noise and harsh environments common to the industrial environment. Additional consideration is required for the selection of materials in special applications such as robotic and welding applications etc.

8-7.2.2 Industrial EtherNet/IP Active Interfaces

Copper and fiber-based Industrial EtherNet/IP products shall meet all applicable requirements of the EtherNet/IP specification, chapter 9. For a product to achieve the Industrial EtherNet/IP performance in harsh environments level, the physical layer shall conform to the requirements as outlined in section 8-9 of this chapter.

8-8 COMMERCIAL Based EtherNet/IP Products and Physical Layer

The use of COTS components may degrade system performance. Careful consideration should be given to the use of COMMERCIAL components in industrial control applications

8-8.1 Copper Media

8-8.1.1 Cables

The transmission performance of shielded or unshielded 4 pair twisted pair cables shall meet the requirements of ANSI/TIA/EIA-568-B.2 standards and the requirements of E1 columns of Table 8-9.4, Table 8-9.5 and Table 8-9.6.

8-8.1.2 Connectors

8-8.1.2.1 RJ-45 Connector Variant

The RJ-45 connectors are the de-facto standard for Ethernet systems. RJ-45 connectors shall meet the requirements stated in ANSI/TIA/EIA-568-B.2. In addition IEC 60603-7 series defines the mechanical and electrical requirements for the RJ-45 connectors.

8-8.1.3 Length Constraints

The total permanent link length for twisted pair systems is limited to 90m (295 ft). The permanent link shall conform to ANSI/TIA/EIA-568-B.1.

The total channel length for twisted pair systems is 100m (328 ft) including patch cables as defined in ANSI/TIA/EIA-568-B.1. Channel and patch cable design and testing shall be in accordance with ANSI/TIA/EIA-568-B.1 and ‘B.2 respectively.

8-9 Industrial EtherNet/IP Media and Physical Layer

8-9.1 Environmental Requirements

Copper and fiber based Industrial EtherNet/IP products should meet the minimum environmental recommendations as defined in Table 8-9.1. Copper and Fiber cabling components shall support the requirements in of Table 8-9.2. Active devices shall meet the minimum EMI requirements of Table 8-9.2. The values found in Table 8-9.2 represent the minimum requirements for IEC light industrial.

Table 8-9.1 Minimum Environmental Recommendations

Environmental Test	Criteria	Industry Standard
Vibration (Unpackaged)		
Frequency Range	10-57Hz	IEC 60068-2-6
Displacement	0.3 mm	
	57-500Hz	
Acceleration	2g	
Shock (Unpackaged)		

Environmental Test	Criteria	Industry Standard
Acceleration	15g (operational)	IEC 60068-2-27
	30g (non-operational)	
Temperature		
Operating range	0 °C min. to +60 °C min. *	IEC 60068-2-1 IEC 60068-2-2
Storage	-40 to +70 °C	IEC 60068-2-1 IEC 60068-2-2
Humidity operating		IEC 60068-2-30
	5 to 95% RH condensing	
Ingress protection		
	IP 20 minimum	IEC 60529
Voltage proof (connector only)		IEC 60512-1
Contact/contact	1000 Vd.c. or a.c. peak	
Contact/test panel	1500 Vd.c. or a.c. peak	
* There may be components or topology de-rating for temperatures below 0 degrees C, or above 60 degrees C.		

Table 8-9.2 Minimum EMI Requirements for EtherNet/IP Components

Environmental Test	Criteria	Industry Standard
EMI		
ESD	4kv contact 8kv air	IEC 61000-6-2 IEC 61131-2 IEC 61326-1
Radiated RF	10V/m @ 80-1000MHz @ 1kHz 3V/m @ 1.4-2.0GHz @ 1kHz 1V/m @ 2.0-2.7GHz @ 1kHz	IEC 61000-4-3
Conducted RF	10V RMS @ 150kHz-80MHz @ 1kHz	IEC 61000-4-6
EFT Comms to ground	2kv	IEC 61000-4-4
Surge Comms to ground	2kv	IEC 61000-4-5
Magnetic field (50/60Hz)	30A/m, 1 min.	IEC 61000-4-8

8-9.2 Copper Media

8-9.2.1 Copper Media Attachment (Normative References)

A copper media attachment to an EtherNet/IP network shall support shielded and unshielded twisted pair technology. The specifications shall contain enhancements (where needed) based on ANSI/TIA/EIA-568-B.1 category 5e cabling performance levels minimum. The signaling and coupling of these variants shall comply with the requirements of IEEE 802.3, 2005 Ed/TP-PMD standard subject to the deviations listed in this section 8-9.2.4. Likewise, the cable's electrical mechanical and environmental performance shall be as defined in section 8-4. The IEEE 802.3 standard defines many internal interfaces within the physical layer. EtherNet/IP products need not directly implement each of these interfaces, but shall behave as if these interfaces exist. These interfaces may be internal to the node and possibly internal to a semiconductor device.

This standard supports 10BASE T and 100BASE TX copper variants as defined by IEEE Std 802.3, 2005 Ed. and the ANSI X.3.263 TP-PMD. Two pair cabling does not support 100BASE-T4 and is infrequently used, therefore 100BASE-T4 is not supported by this standard.

Active devices shall be fitted with one of the jacks defined by this chapter. Attached cables with flying leads or flying leads with jacks are not allowed.

8-9.2.2 Copper Cabling Commercial and Industrial

The cable is critical in influencing the performance of the network in the presence of high noise. To support industrial information and industrial control systems two basic cable types (Commercial and Industrial EtherNet/IP Cables) are recognized. Only cables adhering to this specification will be eligible for the appropriate conformance check mark.

Cables shall conform to the specifications table below.

Table 8-9.3 Minimum Cable Requirements for Commercial and Industrial cabling

Industrial EtherNet/IP Cable Specifications and Requirements					
Specification	**Type**				
Electrical	**Shielded**				**Unshielded**
Conductors	2 or 4 pairs + Shield				2 or 4 pairs
Attenuation Solid Conductors	ANSI/TIA-EIA 568-B.2 Cat 5e Horizontal				ANSI/TIA-EIA 568-B.2 Cat 5e Horizontal
Attenuation Stranded Conductors	ANSI/TIA-EIA 568-B.2 Cat 5e Patch [1]				ANSI/TIA-EIA 568-B.2 Cat 5e Patch [1]
Impedance (fitted) ASTM 4566	95-110 Ω 1-4 MHz 95 – 107 Ω 4-100 MHz				95-110 Ω 1-4 MHz 95 – 107 Ω 4-100 MHz
RL (dB)	1-10 MHz 20 + 6Log$_{10}$(f) 10-20 MHz 26 20-100 MHz 26-5*Log$_{10}$(f/20)				1-10 MHz 20 + 6Log$_{10}$(f) 10-20 MHz 26 20-100 MHz 26-5*Log$_{10}$(f/20)
NEXT Loss (dB)	ANSI/TIA-EIA 568-B.2 Cat 5e				ANSI/TIA-EIA 568-B.2 Cat 5e
Coupling Attenuation (dB)	Freq (MHz)	E1	E2	E3	n/a
		See Table 8-9.5			
Shielding Effectiveness	tbd				n/a

Industrial EtherNet/IP Cable Specifications and Requirements					
Specification	**Type**				
Electrical	**Shielded**	**Unshielded**			
Capacitance unbalance	<= 150pf/100meter	< = 150pf /100meter			
DCR	9.38 Ω/100 meters	9.38 Ω/100 meters			
DCR Unbalance	3%	3%			
TCL	n/a	Frequency (MHz)	E1	E2	E3
			See Table 8-9.4		
ELTCTL		Frequency (MHz)	E1	E2	E3
			See Table 8-9.5		
Mechanical	**Shielded**	**Unshielded**			
Pulling Tension	111 N	111 N			
Breaking Strength	400 N	400 N			
Bend Radius	1" at -20C	1" at –20C			
Dimensional (Recommended for RJ 45 compatibility)	**Shielded**	**Unshielded**			
Jacket OD	0.315" Max	0.315" Max			
Insulated Conductor	0.048" Max	0.048" Max			

1 The insertion loss is based on COMMERCIAL cables. Other constructions, such as high flex, may have different performance. Consult the manufacturer for more information.

8-9.2.2.1 Cabling Balance

8-9.2.2.1.1 Unshielded twisted pair transverse conversions loss (TCL) and equal level transverse conversion transfer loss (ELTCTL)

Each pair of unshielded twisted-pair channels shall meet the TCL requirements of Table 8-9.4 and ELTCTL requirements of Table 8-9.5 below. TCL and ELTCTL shall be measured in accordance with ANSI/TIA/EIA-568-B.2-9.

Table 8-9.4 TCL Limits for Unshielded Twisted Pair Cabling

Category	Frequency (MHz)	Minimum TCL (dB) ISO/IEC 24702		
		E_1	E_2	E_3
5e	$1 \le f < 30$	53-15log(f), (40 max)	63-15log(f), (40 max)	73-15log(f), (40 max)
	$30 \le f \le 100$	60.4 -20log(f)	70.4 -20log(f)	80.4 -20log(f)

Table 8-9.5 ELTCTL Limits for Unshielded Twisted Pair Cabling

Category	Frequency (MHz)	Minimum ELTCTL (dB) ISO/IEC 24702		
		E_1	E_2	E_3
5e and 6	$1 \le f \le 30$	30-20log(f)	40-20log(f)	50-20log(f), (40 max)

8-9.2.2.1.2 Shielded Twisted Pair Coupling Attenuation

Each pair of screened twisted-pair channels shall meet the coupling attenuation requirements of Table 8-9.6.

Table 8-9.6 Coupling Attenuation for Screened Twisted Pair Cabling

Category	Frequency (MHz)	Minimum Coupling Attenuation (dB) ISO/IEC 24702		
		E_1	E_2	E_3
5e	30 ≤ f ≤100	40	50	60
6	30 ≤ f ≤ 250	80-20log(f) (Max 40 dB)	90-20log(f) (Max 50 dB)	100-20Log(f) (Max 60 dB)

Note for EMC purposes, coupling attenuation should be measured up to 1 GHz

Coupling attenuation shall be measured in accordance with IEC 61156-5

8-9.2.2.1.3 Two and four pair color codes

Two and four pair cable color codes shall be as defined Table 8-9.7 and Table 8-9.8 in respectively

Table 8-9.7 Two Pair Color Codes

Pair Assignment	Signal Name	2 Pair
Pair 1	TX+	White-orange
	TX-	Orange
Pair 2	RX+	White-green
	RX-	Green

Table 8-9.8 Four Pair Color Codes

TIA Pair Assignment	Signal Name	Color
Pair 2	TX+	White-orange
	TX-	Orange
Pair 3	RX+	White-green
Pair 1[1]	NA	Blue
	NA	White-blue
Pair 3	RX-	Green
Pair 4 [1]	NA	White-brown
	NA	Brown

1 Not used for 10 Mbps and 100 Mbps TX networks

8-9.2.3 Connectors

8-9.2.3.1 Industrial EtherNet/IP Connector RJ-45 Variant

Attachment to the medium shall be via either of two types of Industrial grade RJ-45 connectors:

- Non-Sealed industrial RJ-45 EtherNet/IP connector – The RJ-45 EtherNet/IP connector shall meet the IEC 60603-7 standard and additional requirements of this chapter.
- Sealed Industrial EtherNet/IP RJ-45 connector housing – The IP67 sealed industrial EtherNet/IP connector housing shall conform to the specifications IEC 61076-3-106.

8-9.2.3.1.1 Sealed and Non-Sealed Industrial EtherNet/IP Connector

Standard industrial hardened RJ-45 connector shall meet the following specifications:

Industrial EtherNet/IP Connector Specifications and Requirements		
Specification	**Type**	
Electrical	**RJ-45-Shielded**	**RJ-45**
Conductors	8 + 1 Shield	8
Insertion Loss	ANSI/TIA/EIA-568-B.2 Category 5E	ANSI/TIA/EIA-568-B.2 Category 5E
RL	ANSI/TIA/EIA-568-B.2 Category 5E	ANSI/TIA/EIA-568-B.2 Category 5E
NEXT Loss	ANSI/TIA/EIA-568-B.2 Category 5E	ANSI/TIA/EIA-568-B.2 Category 5E
Shielding Effectiveness	ANSI/TIA/EIA-568-B.2 Category 5E	N/A

Mechanical	**RJ-45-Shielded**	**RJ-45**
Gender	Plug and Socket	Plug and Socket
Mating Specification	CEI IEC 60603-7	CEI IEC 60603-7
Contact plating	50u inches min. gold over 100u inches min. nickel or equivalent plating system	50u inches min. gold over 100u inches min. nickel or equivalent plating system
Contact LLCR over life	< 20 mΩ	< 20 mΩ
Initial Contact Low Level Contact Resistance	<=2.5 mΩ	<=2.5 mΩ
Minimum contact force	100 grams	100 grams
Minimum plug retention force [1]	133 N	133 N
Contact Life	750 insertions and extractions min.	750 insertions and extractions min.

1 Required when the connector is used as a standalone connector (not in a protective shell)

The non-sealed connector shall be wired in accordance with the pin/wire assignments in Table 8-9.9.

Table 8-9.9 8 Way Modular Connector Pin/Pair Cable Assignment

PIN	Signal Name	Pin T568A	Pair Assignment	Pin T568B	Pair Assignment
1	TXD+	White Green	Pair 3	White Orange	Pair 2
2	TXD-	Green		Orange	
3	RXD+	White Orange	Pair 2	White Green	Pair 3
4	NA[1]	Blue	Pair 1	Blue	Pair 1
5	NA[1]	White Blue		White Blue	
6	RXD-	Orange	Pair 2	Green	Pair 3
7	NA[1]	White Brown	Pair 4	White Brown	Pair 4
8	NA[1]	Brown		Brown	

1 Not used for 10 Mbps and 100 Mbps Networks

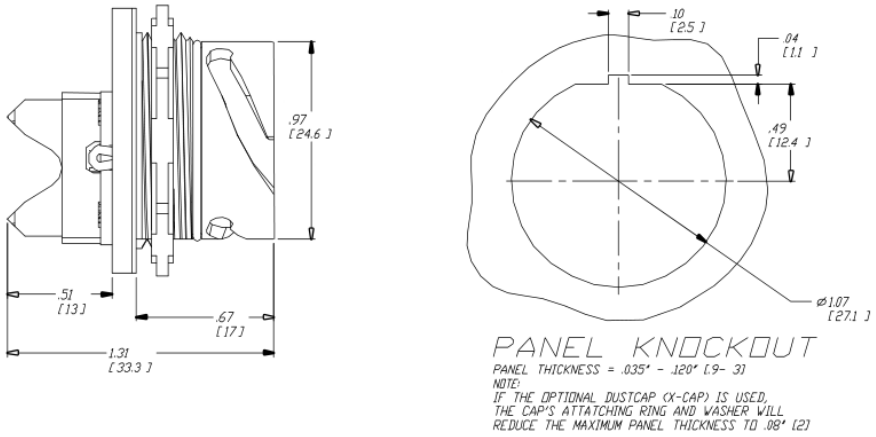
Both ends of the cable shall be wired the same unless constructing a crossover cable.

8-9.2.3.1.2 Sealed Industrial EtherNet/IP RJ-45 Housing

The sealing interface shall meet a minimum of IP67 sealing performance as defined in IEC 60529. The pin/pair wiring of section 8-9.2.3.1.1 applies to the Sealed Industrial EtherNet/IP 8-Way modular connector. Cross over cable are allowed within the same connector family.

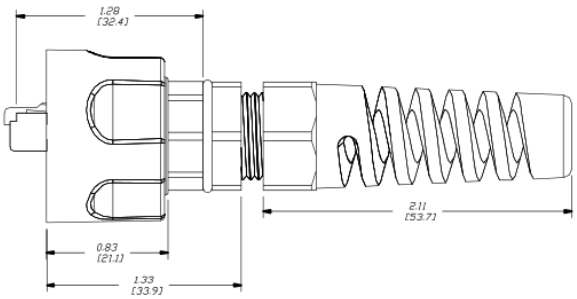
The Sealed RJ-45 variant 1 is based on the IEC 61076-3-106 specification. The following sealed jack drawing sufficiently defines the jack to maintain compatibility for mating and sealing amongst various vendors who may make one or both parts. The jack may be offered as a PCB mount, bulkhead connector and cable end either field installed or manufactured assembly. The jack is fully compatible with standard off-the-shelf plugs.

Figure 8-9.1 Typical Sealed Jack



The Sealed RJ 45 variant 1 is based on the IEC 61076-3-106 specification. The following sealed plug drawing sufficiently defines the plug to maintain compatibility for mating and sealing amongst various vendors who may make one or both parts. The plug may be offered as a field installable or manufactured cable assembly. The plug housing will accommodate a standard plug as defined by IEC 60603-7 standard with the exception of the locking mechanism, which is disabled.

Figure 8-9.2 Typical Sealed Plug



8-9.2.3.1.3 Sealed M12-4 “D” Coding

The M12-4 “Type D” coding connector is well known and accepted in industrial ethernet applications – for more than 20 years it has been the standard for connection of sensors in the industry. The connector is defined in Amendment 1 to IEC 61076-2-101, 4-pin “Type D” Coding. The 4-pin M12 connector is suitable for use with 2-pair shielded or unshielded Ethernet cables only.

The M12-4 “D” coding connector shall be wired in accordance with the pin/wire assignments in Table 8-9.10.

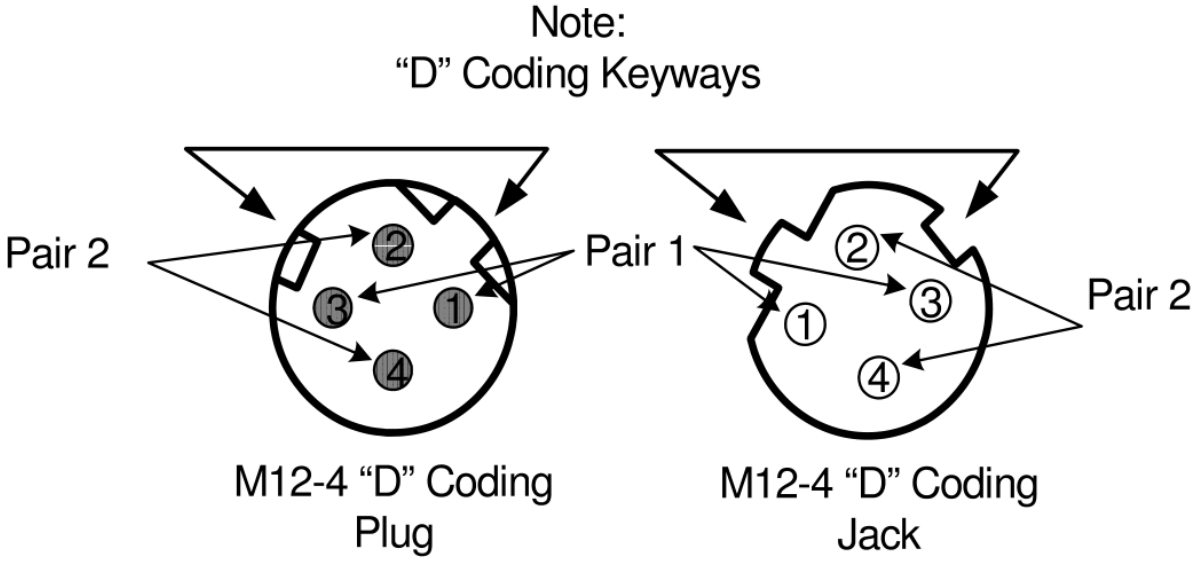
Table 8-9.10 M12-4 "D" Coding Cable Pin/Pair Cable Assignment

PIN	Signal Name	Color Code	Pair Assignment
1	TXD +	White Orange	Pair 1
3	TXD -	Orange	
2	RXD +	White Green	Pair 2
4	RXD -	Green	

Construction of crossover cables and conversion cables is permissible. Conversion cables constructed with 4-circuit M12-4 “D” coding connectors to 8-Way modular connectors shall be constructed from 2 pair cables containing the wire color codes defined in Table 8-9.10. The use of a 4 pair cable with 4 position M12-4 “D” coding connector is not permissible.

The 4-Pin M12 connector is suitable for use with 2 pair shielded or unshielded Ethernet cables only.

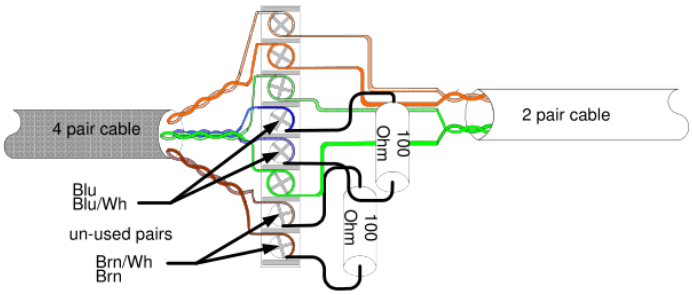
Figure 8-9.3 Plug Side and Jack Side Mating View



8-9.2.3.2 Mixing 2 and 4 Pair Cabling Components in a Channel

Cords using multi family connectors are permissible provided the number of conductors in the cable is equal to the minimum contact number of connector of the least contact assignment. For example, 4 pair cables shall not be used in the same channel with M12-4 "D" coding connectors. An exception to this requirement is where the unused pairs of active channel are terminated at their characteristic impedance. If terminated, a differential termination without reference to ground is preferred. Figure 8-9.4 shows the concept of terminating the un-used pairs of active channels in a 4 pair cable. The use of terminal strips is not recommended and is only here for illustration. Termination is required at both ends of the active cable where the pairs are not used.

Figure 8-9.4 Example of termination of un-used pair (for reference only)



8-9.2.3.3 Coupler

A coupler consists of two closely spaced (less than 10cm) electrically connected interfaces. Both interfaces are of the same physical mating interface.

A mated coupler shall conform to the transmission requirements of one connection of the appropriate media and category.

If the interfaces are not electrically close or the coupler does not meet the transmission of the appropriate media and category, then the coupler shall be counted as two mated connections.

8-9.2.3.4 Adapter

An adapter consists of two closely spaced electrically connected interfaces. They may be of different circuit counts. Both interfaces may be of the same or different physical mating interface; for example an M12-4 "D" coding connector to a RJ-45.

A mated adapter shall conform to the transmission requirements of one connection of the appropriate media and category.

If they not electrically close or the adapter dose not meet the transmission of the appropriate media and category, then the adapter shall be counted as two mated connections.

8-9.2.3.5 Bulkhead Connectors

Bulkhead connectors are typically used at environmental or enclosure boundaries to facilitate connection and disconnection to the enclosure. A term used to define a mounting style of connectors. Bulkhead connectors are designed to be inserted into a panel cut-out from the rear (component side) or front side of the panel. The connector should be used where cables enter or exit the cabinet to maintain enclosure seal integrity. In addition they may be used to construct modular systems whereby providing modular connectivity.

Bulkhead connectors allow systems to be designed and built in modular configurations. This method should be considered based on user design and service preferences. Modularity provides quick deployment and ease of serviceability.

The designer shall be aware of metallic bulkhead feed throughs that connect the cabling at the enclosure wall. This may form a ground loop that could disrupt communications. Where a ground loop may be formed, a separate grounding conductor should be installed to provide an equal potential between the two points. An alternative method would be to isolate the bulkhead feed through using an insulator between the bulkhead feed through and the enclosure wall.

The transmission performance requirements for a bulkhead connector are defined in section 8-9.2.3.5. Figure 8-9.5 is an example of M12-4 D-coding EtherNet/IP bulkhead feed through connectors.

Figure 8-9.5 M12-4 to 8-way Modular Bulkhead

Consult the manufacturer’s data sheet for mounting hole cut out dimensions. Consider the panel minimum and maximum wall thickness of the enclosure when selecting a bulkhead.

8-9.2.3.6 Industrial Channel Length

8-9.2.3.6.1 Patch Cord Length

EtherNet/IP specifications limit the channel to 100 meters or up to 90 meters horizontal wiring with two 5-meter patch cords. Some applications will require longer patch cords. In these applications the total length of horizontal wiring must be adjusted to compensate for the added loss of each connector pair and additional patch cord length beyond 10m.

$$C = \frac{(102 - H)}{(1 + D)} \qquad (1)$$

Where:

C is the maximum combined length (m) of the work area cable, equipment cable, and patch cord.

H is the length (m) of the horizontal cable (H + C </= 100 m).

D is a de-rating factor for the patch cord type (0.2 for 24 AWG UTP/24 AWG ScTP and 0.5 for 26 AWG ScTP). The de-rating factors are based on COMMERCIAL cables. Other constructions, such as high flex, may have different performance. Consult the manufacturer for more information.

W is the maximum length (m) of the work area cable

T is the total length of horizontal, patch and equipment cords.

The maximum stranded cable length is limited to 85m for the channel with the standard 20% derating for standard stranded cables.

Table 8-9.11 Wire Type versus Length

	D	H	W	C	T
Patch Cable Gauge	**Patch Derating**	**Horizontal Length, (H+C<=100 m)**	**Patch Length**	**Total Length Patch and Equipment**	**Total length of patch, equipment and horizontal**
#24	0.2	100	0	0	100
#24	0.2	0	80	85	85
#24	0.2	25	59	64	89
#24	0.2	50	38	43	93
#26	0.5	0	63	68	68
#26	0.5	25	46	51	76
#26	0.5	50	30	35	85
#26	0.5	100	0	0	100

8-9.2.3.6.2 Channel Length Based on Temperature

Elevated temperatures cause higher signal loss in copper cables due to increased resistance. This added loss must be considered in addition to the type of copper cable (solid conductor horizontal or stranded conductor patch) to determine the maximum channel length. Shielded (STP) copper cable typically exhibit 0.2% attenuation increase for every 1° C temperature rise above 20° C to 60° C. Unshielded (UTP) Category 5e cables typically exhibit 0.4% attenuation increase for every 1° C temperature rise from 20° C to 60° C. Unshielded (UTP) Category 6 cable exhibit 0.4% attenuation increase for every 1° C temperature rise from 20° C to 40° C and 0.6% attenuation increase for every 1° C temperature rise from 40° C to 60° C, due to more copper and plastic content. The elevated temperature insertion loss is based on COMMERCIAL cables. Other constructions, such as high flex, may have different performance. The change in attenuation with temperatures beyond 60° C is product specific. Consult your supplier for more information.

The channel length and attenuation are linearly related, that is a 12% increase in attenuation reduces the channel length 12%. The following examples show how to calculate the maximum channel length for a given configuration and temperature.

AElev.Temp.=AIncrease Coefficient * Δ T

LElev.Temp.=AIncrease Coefficient * Δ T

Where: AElev.Temp = elevated temperature attenuation
AIncrease Coefficient = attenuation temperature coefficient
Δ T = change in temperature
LElev.Temp = elevated temperature maximum length

Assume you want to use solid conductor, Category 5e, horizontal cable at 60° C.

Note: The entire length should be treated as if the temperature is the worst-case temperature to ensure a conservative, simplified calculation.

You are limited to 100 meters based on the cable type. This distance must be de-rated to accommodate the elevated temperature. 60° C is 40° C above 20° C. 40° C times 0.4% equals 16% length reduction. The length reduction is calculated by taking the percent reduction times the cable type length limit: 16% x 100 meters = 16 meters.

The maximum channel length is calculated by subtracting the elevated temperature length reduction from the cable type channel limit: 100 meters – 16 meters = 84 meters. The maximum channel length for all solid, horizontal Cat 5e cable at 60° C is 84 meters.

For all stranded conductor patch Cat 5e at 60° C we have the following:

Cable type channel limit= 85 meters
Temperature change = 40C
Temperature coefficient = 0.4%
Total change = 16%
Length reduction = 13.6 meters
Maximum channel length for all stranded, patch Cat 5 at 60° C is 68.7 meters.

For 25 meters solid, horizontal Cat 5e cable with some length of #24 AWG, stranded conductor, Cat 5e patch at 40° C we have the following:

- 25 meters of solid, horizontal cable at 40° C has the loss of 8% more length of cable, 25 x 1.08 = 27 meters effective length
- Based on 27 meters we can have the effective length of patch as, (102-27)/(1+0.2)=62.5
- Total effective maximum stranded, patch length = 62.5 meters
- 62.5 meters of stranded, Cat 5e patch has 8% more loss then the actual length at 20° C, 62.5/1.08 = 57.9 meters actual length.
- The actual maximum stranded length = 57.9 meters
- The total channel length limit is the sum of the actual solid, horizontal cable maximum length limit plus the actual stranded, patch cable maximum length limit, 25 + 57.9 = 82.9 meters
- The maximum channel length limit for 25 meters of solid conductor, horizontal Cat 5e cable is 82.9 meters at 40° C with a maximum of 57.9 meters of stranded conductor, Cat 5e patch cable.

8-9.2.3.7 Industrial Permanent Link

The length of a industrial permanent link is limited to that of 8-9.2.3.6 less the equivalent length of 10 meters of patch cords.

8-9.2.3.8 Number of connections in a channel:

The number of mated connections allowed in a channel is determined by the desired channel performance (Category) and the performance level of the components selected. A Mated Connection is defined as an electrically conductive communications path comprised of a mated jack and plug. Back to back jack bulkheads may be counted as one connection provided they meet the requirements of this chapter. Cable lengths between connecting hardware greater than 10cm shall be counted in the total channel/link appropriate cable length budget.

Alternate configurations should be field tested to ensure adequate performance. Table 8-9.12 provides guidance for connector cable performance levels to achieve a given category channel for more than 4 connections.

Table 8-9.12 Number of allowable Connections in a Channel

Desired Channel performance	Number of Mated connections	Category connector (required)	Category cable (required)
5e	6	6A	5e

Current studies show that:

a) A Category 5e channel topology can include up to 6-mated connections, where each mated connection meets minimum Category 6A performance.

b) Maximum distance between jack and jack of the bulkhead connection is 10 cm. If the distance is greater than 10 cm each plug/jack interface shall be considered as a separate mated connection.

In order to maintain Category 5e performance in the channel for more than 4 mated connections, Category 6A connections shall be used. See Table 8-9.13 for return loss and NEXT, transmission requirements for construction of higher count channels.

Table 8-9.13 Transmission Requirements for More Than 4-connections in a Channel

Desired Channel Class	Number of Connections	Required Minimum Connecting Hardware Return Loss (dB)	Required Minimum Connecting Hardware NEXT (dB)	Cable Category
5e	5 or 6	26-20 log(f/100)	54-20log(f/100)	CAT 5e

8-9.2.3.9 Bulkhead Feed Through and Cable Glands

8-9.2.3.9.1 Bulkhead Cable Glands

Bulkhead cable glands provide entry/exit passages for permanently installed cables. Bulkhead feed troughs and/or bulkhead connectors allow systems to be designed and built in modular configurations. This method should be considered based on user design and service preferences. Modularity provides quick deployment and ease of serviceability.

8-9.2.3.9.2 Channels Using Balanced Cabling Bulkhead Connections

Figure 8-9.6 shows an intermediate cabling channel and a floor distribution channel created using a fixed cable terminated at a closure bulkhead.

The length of the fixed cable used within a channel shall be determined by the equations shown in section 8-9.2.3.6.

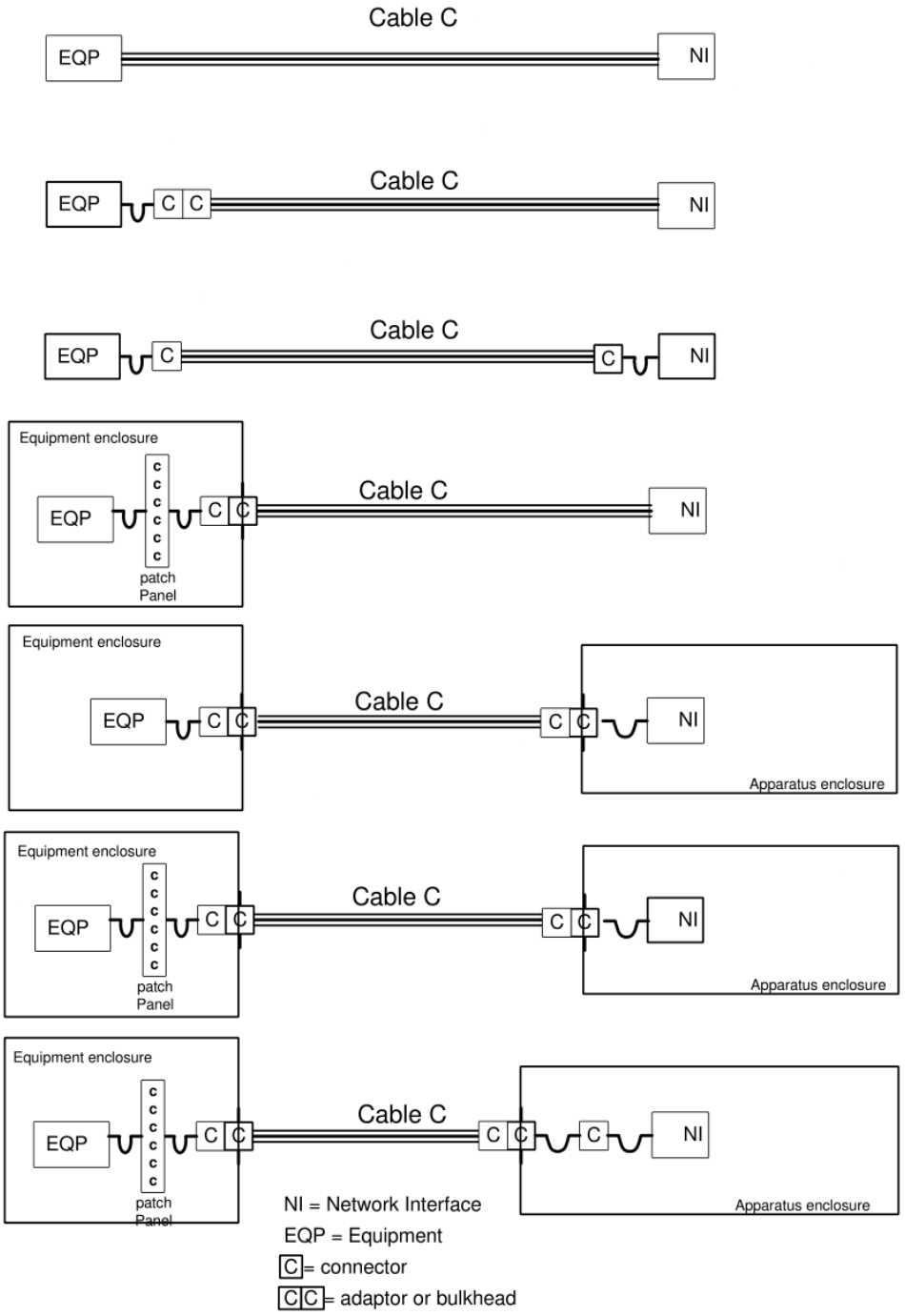
In section 8-9.2.3.6, it is assumed that;

a) The flexible cable within these cords has a higher insertion loss specification than that used in the fixed cable,

b) The cables within these cords in the channel have a common insertion loss specification.

The maximum length of the fixed cable will depend on the total length of cords to be supported within a channel. During the operation of the installed cabling, a management system should be implemented to ensure that the cords used to create the channel conform to the design rules for the floor, building or installation.

Figure 8-9.6 Channel Configurations

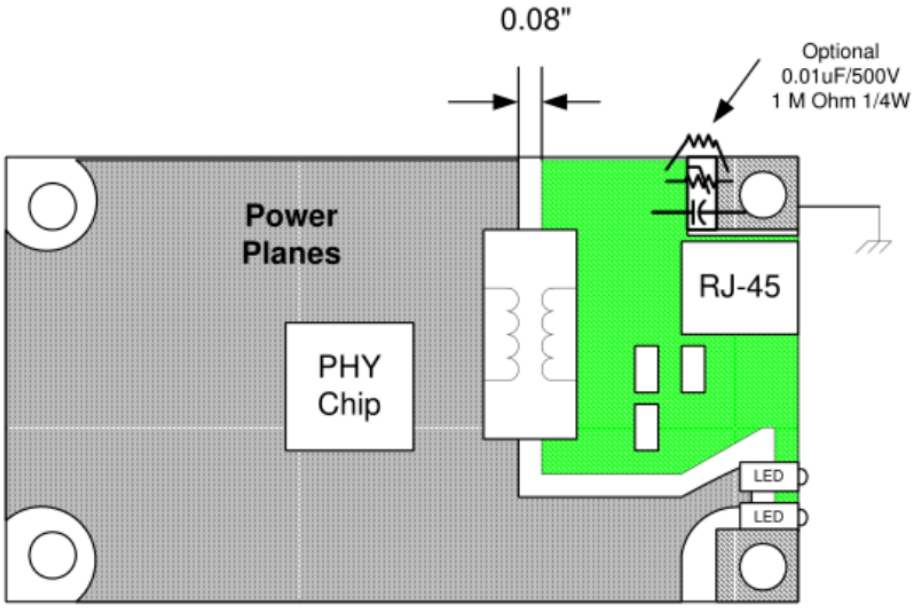


8-9.2.4 Industrial EtherNet/IP TP-PMD (Normative References)

A device that connects to the Industrial EtherNet/IP copper media shall conform to IEC 802.3 and ANSI X3.263 TP-PMD standard unless noted in this subclause.

The impedance at the media interface shall conform to ISO/IEC 802.3 (ANSI/IEEE Std 802.3) and IEEE Std 802.3u-1995 supplement with the exception of impedance tolerance. The temperature range and vibration shall be consistent with the targeted environment. In some cases it may be necessary to add components to protect the PMD from surge, ESD, EFT and conducted noises. Figure 8-9.10 is an example of how protection devices may be used to protect the EtherNet/IP device. In order to maximize the performance in noise, it is critical that the components selected for the PMD provide key characteristics. The transformer should (highly recommended) provide a minimum of 59dB common mode rejection (CMR) at 30 MHz. In addition special care in the circuit board trace parameters is needed to maintain impedance and noise immunity. Figure 8-9.7 is an example layout showing ground planes and isolation areas to help maintain noise immunity.

Figure 8-9.7 Example Reference Circuit Board Layout (informative)



A copper media attachment to an EtherNet/IP network shall support shielded and unshielded twisted pair technology. Active interfaces shall be compatible with ANSI/TIA/EIA-568-B.1 category 5e cabling system and cabling/component enhancements specified by this chapter. The signaling, encoding and coupling of these variants shall comply with the requirements of IEEE 802.3/TP-PMD and ANSI X3.263 TP-PMD standard subject to the deviations listed in this chapter. Likewise, the cable's electrical mechanical and environmental performance shall be as defined in section 8-9.1. The environmental classifications that support this requirement is defined the MICE table as defined by IEC 24702. The IEEE 802.3 standard defines many internal interfaces within the physical layer. EtherNet/IP products need not directly implement each of these interfaces, but shall behave as if these interfaces exist. These interfaces may be internal to the node and possibly internal to a semiconductor device.

At a minimum active interfaces shall support 10BASE T and 100BASE TX as defined by IEEE Std 802.3, 2000 Ed. and the ANSI X.3.263 TP-PMD. Two pair cabling will not support 100BASE-T4 therefore 100BASE-T4 interfaces are not supported by this standard.

8-9.2.4.1 Network Jacks for Active Devices

Active devices are end devices such as computers, sensors and HMIs. These devices generally support single network connections unless redundant. An active device with an embedded switch shall support AutoMDIX on its ports and be wired in accordance with this sub chapter. Repeaters should default to MDIX mode when AutoMDIX or Auto Negotiation is disabled. Active devices shall be fitted with one of the jacks defined in this chapter. Attached cables with flying leads or flying leads with jacks are not allowed. The jacks for active devices shall be wired in accordance with the pin definition described in Table 8-9.9 and Table 8-9.10.

8-9.2.4.2 Network Jacks for Connectivity Devices (repeaters)

Connectivity devices are active devices used to control the flow of data throughout the infrastructure. For example connectivity devices are classified as repeaters, switches, routers and bridges. These devices may have one or more of the following ports, LAN, WAN, Uplink. Figure 8-9.8 shows the relationship between LAN and WAN/Uplink ports for connectivity and active devices. Connectivity devices such as Switches, routers and bridges shall be fitted with jacks. The LAN side of the connectivity devices shall be wired in accordance with Table 8-9.14 and Table 8-9.15 or provide AutoMDIX. If the connectivity device supports AutoMDIX, then it shall default to MDIX state when AutoMDIX is disabled. WAN ports including uplink ports shall be wired in accordance with Table 8-9.9 and Table 8-9.10.

Figure 8-9.8 Port Identification

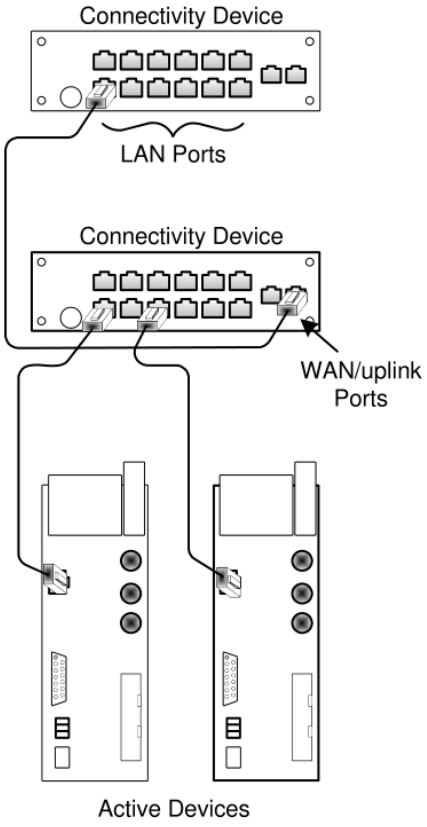


Table 8-9.14 8-Way Modular Jack Pin Assignment for LAN Ports

PIN	Signal Name
1	RXD+
2	RXD-
3	TXD+
4	NA[1]
5	NA[1]
6	TXD-
7	NA[1]
8	NA[1]

Table 8-9.15 M12-4 "D" Coding Jack Pin Assignment for LAN Ports

PIN	Signal Name
1	RXD +
3	RXD -
2	TXD +
4	TXD -

8-9.3 Termination for a 10/100 Mbps Interface with 4 Pair Support

Active devices shall use an appropriate termination technique such as found in Figure 8-9.9 for both the used and unused pairs. The unused pairs shall be terminated into their characteristic impedance at the device to prevent reflections of coupled energy. A common mode termination shall be used to terminate the TXD and RXD pairs. The resistor values may be adjusted between 50Ω and 75Ω to obtain 100Ω differential impedance and the appropriate common mode impedance.

Figure 8-9.9 PHY of Termination Example

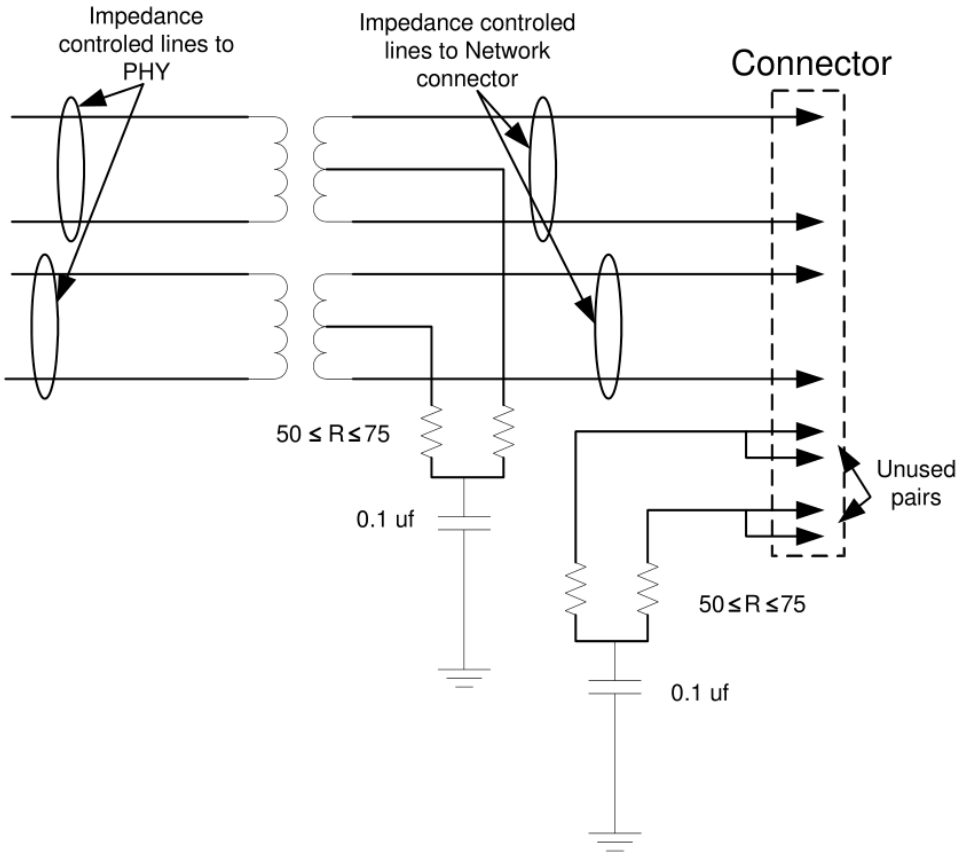
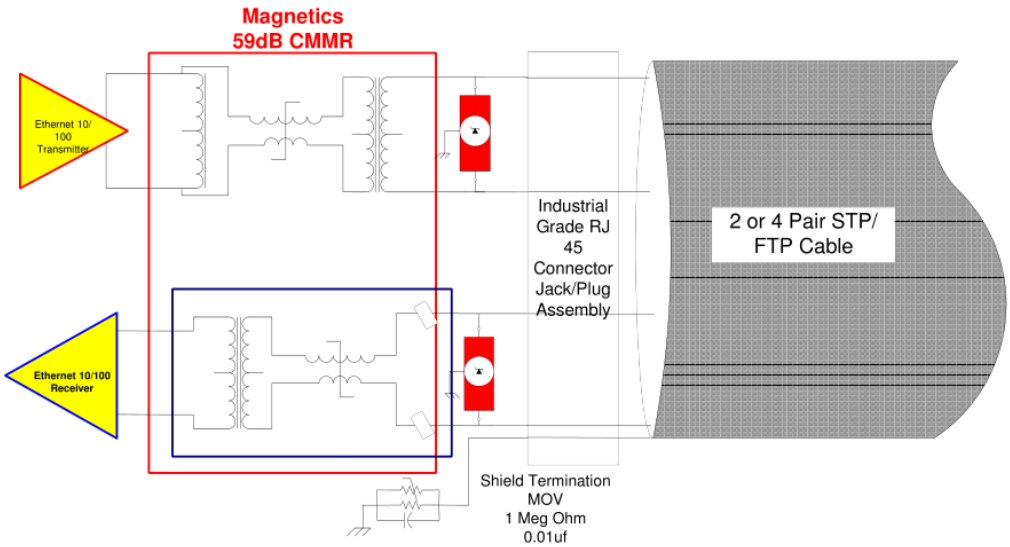


Figure 8-9.10 Example Physical Layer Block Diagram (informative)



8-9.4 Shield Grounding

8-9.4.1 Connectivity Device (Switch, Hub, Bridges, Routers, etc.)

The communications shield shall be terminated directly to earth ground in accordance with IEEE 802.3.

8-9.4.2 Two Port Devices

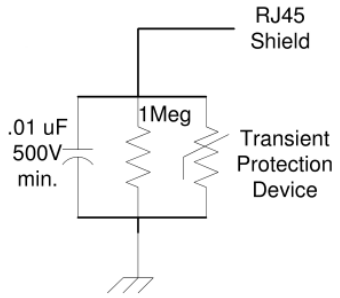
Two port devices that provide two active ports should not use ganged RJ 45 jacks with common shields; doing so will propagate the grounds and potentially cause ground loops in the system. Termination of the shield shall be in accordance with Active Devices.

8-9.4.3 Active Devices (sensor, PLC etc.)

To prevent ground loops caused by shielded cables, devices shall not connect the shield directly to ground. Industrial EtherNet/IP devices shall provide shield terminated as detailed in Figure 8-9.11. For Commercial active devices where the shielded RJ 45 connector provides direct ground, the shield should be disconnected at the active device end of the channel as shown in Figure 8-9.12 and Figure 8-9.13.

The shield termination for Industrial EtherNet/IP active devices, using a parallel resistor and capacitor is shown in Figure 8-9.11.

Figure 8-9.11 Shield Termination for Devices



If the active device provides direct connection to ground through the RJ-45 connector, then the shield shall not be connected at the RJ45 plug. Figure 8-9.12 and Figure 8-9.13 are examples of how to break the shield at a device that is directly grounded.

Figure 8-9.12 Example Shield Termination

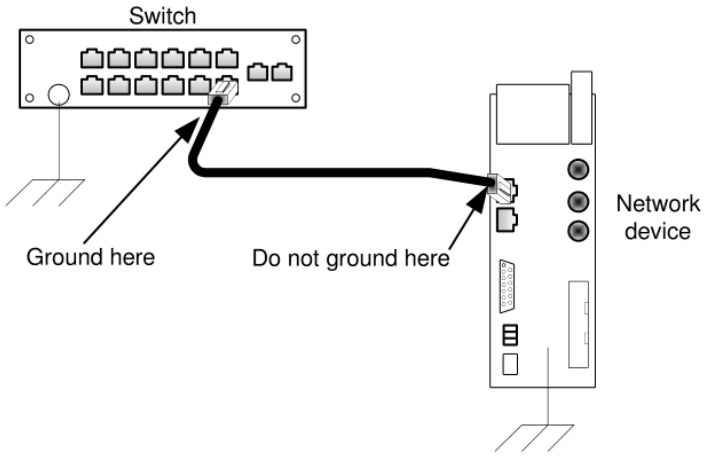
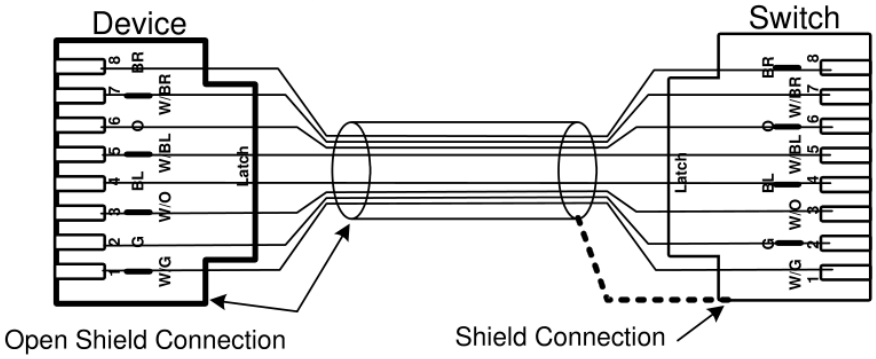


Figure 8-9.13 Shield Termination for Commercial Devices



8-9.5 Fiber Media Variant

8-9.5.1 Cables

The following fiber optic cables are supported by this standard:

Table 8-9.16 Recognized Fiber Cables

Fiber Type	Supported Fiber	Wavelength (typical)
Multimode	50/125µm, 62.5/125µm	1310 nm
Singlemode	9/125 µm	1310 nm
Step Index Multimode	1,000 µm	650 nm

8-9.5.1.1 Multi Mode Fiber Optic Cables

The following multimode fiber optic cables are in accordance with ANSI/TIA/EIA 568-B.3.

- 62.5/125µm
- 50/125µm

8-9.5.1.2 Single mode fiber optic 9/125µm

The single mode fiber shall conform to ANSI/EIA/TIA 568-B.3 standard.

8-9.5.1.3 Step Index Multimode 1mm Polymer Optical Fiber (POF)

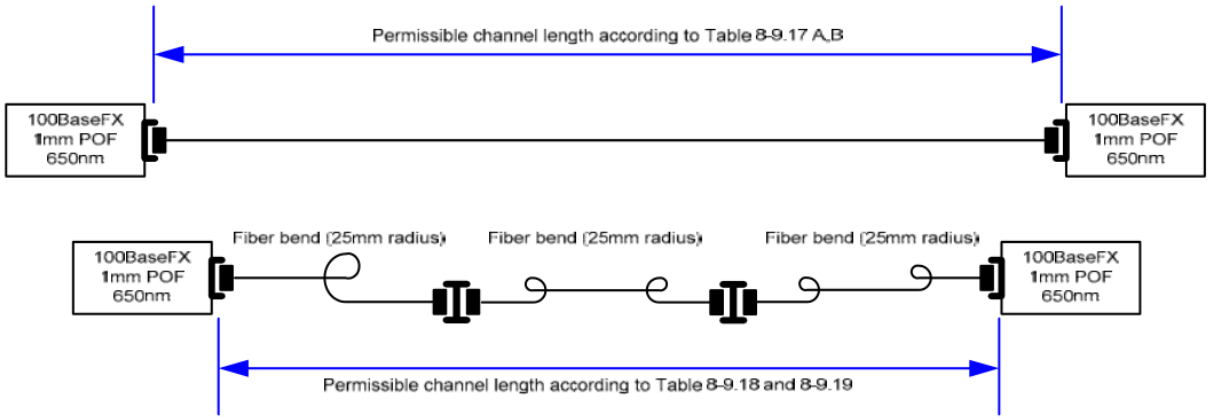
The 1mm POF optical performance and mechanical dimensions of the bare fiber shall conform to requirements of IEC 60793-2-40 Ed2 for category A4 fiber at its minimum requirements. There are two fiber numerical apertures (NA) recognized by this standard as defined below in Table 8-9.16 for A4a.2 and A4d requirements and category. The 1mm POF jacketed cables shall conform to IEC 60794-2-42 Ed1. Simplex and multicore fiber cables are supported by this standard.

Table 8-9.17 Multimode, POF, 1mm Cable Specifications

Fiber Category	Fiber NA	Fiber cable specification	Fiber mechanical specification
A4a.2	1mm POF 0.5 NA	IEC 60794-2-42 Ed1	IEC 60793-2-40 Ed2
A4d	1mm POF 0.3 NA	IEC 60794-2-42 Ed1	IEC 60793-2-40 Ed2
Note: Mixing of numerical apertures within the same channel is not recommended.			

The following figures show a channel with no bends and a channel with multiple bends.

Figure 8-9.14 Maximum Channel Lengths



The number of bends within the channel will reduce the allowable channel length. The channel length shall not exceed the lengths shown in the following tables for the number of bends in the channel. The following tables describing the maximum channel lengths are based on worst case allowable for number of bends and connections at 25ºC.

Table 8-9.18 1 mm A4a.2 POF 0.5 NA

Maximum link length with additional losses due to bend radius and number of connections			**Number of connections (Note 2)**		
			0	1	2
	Worst Case losses (db)		0db	3db	6db
	#	Maximum losses(db)due to bend radius (Note 3)	Maximum length in Meters (Note 1)		
#Bends /Loss	0	0.00	55	43	32
	1	0.87	52	40	29
	2	0.96	51	40	28
	3	1.03	51	40	28
	4	1.06	51	39	28
	5	1.09	51	39	28

Table 8-9.19 1mm A4d POF 0.3 NA

Maximum link length with additional losses due to bend radius and number of connections			**Number of connections (Note 2)**		
			0	1	2
	Worst Case losses (db)		0db	3db	6db
	#	Maximum losses(db)due to bend radius (Note 3)	Maximum length in Meters (Note 1)		
#Bends /Loss	0	0.00	65	53	41
	1	0.62	63	51	39
	2	0.64	62	50	38
	3	0.67	62	50	38
	4	0.69	62	50	38
	5	0.70	62	50	38

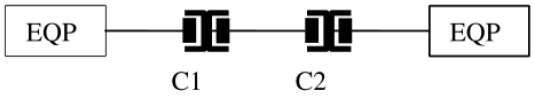
Notes for Table 8-9.18 and Table 8-9.19:

Note 1: Fiber loss

Fiber base loss includes environmental influences mainly from absorption of water (which is reversible), and the Launch NA of the source.

Note 2: Connections

Bulkhead coupling loss is included in minimum launch power of the transmitter and worse case receiver sensitivity. Worst case fiber to fiber loss by a connector is assumed to be 3.0 dB. Two mated connectors coupled with an adapter make a connection.



Note 3: Bending loss

Bend loss is measured for the fiber with maximum link length. The above table Bending losses at 25mm Radius is used as typical value, 1 bend = 360o. If a bend radius is greater than 180 mm it is considered straight with no loss. Other bend radiuses are allowable and consult manufacturer's data sheet for bending losses.

8-9.5.2 Connectors

The fiber media attachment to an EtherNet/IP network shall be limited to the LC, SC, SCRJ, ST and Sealed M12 Fiber Connector variants. The signaling and coupling for the fiber types shall be as specified in the IEEE 802.3 standard subject to the deviations listed in this section (section 8-9.5). The SC and ST connectors are legacy connectors and therefore are allowed by this standard, however are not recommended for new designs. EtherNet/IP devices utilizing the LC transceivers shall have duplex jacks with center spacing compatible with the FOCIS standard of 0.246 inches (6.25mm). Permanently attached fiber pigtails shall not be used.

EtherNet/IP devices utilizing 100BASE-FX medium dependent interfaces (MDI) shall support one of the connectors listed in Table 8-9.20. If the MDI supports POF the MDI shall connect to the Medium via SCRJ, Sealed M12 Fiber optic connector or connector-less transceiver.

8-9.5.2.1 Non-Sealed Connectors

Table 8-9.20 Non-sealed Connector Types and Reference Standards

Non-sealed connector type	Reference Standards
LC, ST, SC	ANSI/TIA/EIA-568-B.3, FOCIS
SCRJ	IEC 61754-24 / EN 50377-6-1

Note: IEC 61754-24 and EN 50377-6-1 is currently at CDV as of 10/10/08.

Table 8-9.21 LC, SC, SCRJ and ST Connector Insertion Loss

Fiber Medium	Connector Loss at Wavelength		Connector(s)
	650nm	1310nm	
9/125µm	Not supported	0.75 dB max.	LC, SC[1], SCRJ and ST[1]
50/125µm	Not supported	0.75 dB max.	LC, SC[1], SCRJ, ST[1], Circular M12
62.5/125µm	Not supported	0.75 dB max.	LC, SC[1], SCRJ, ST[1], Circular M12
1mm POF	1.5 dB max	Not supported	SC, ST and connector-less

1 ST and SC connectors are not recommended for new designs

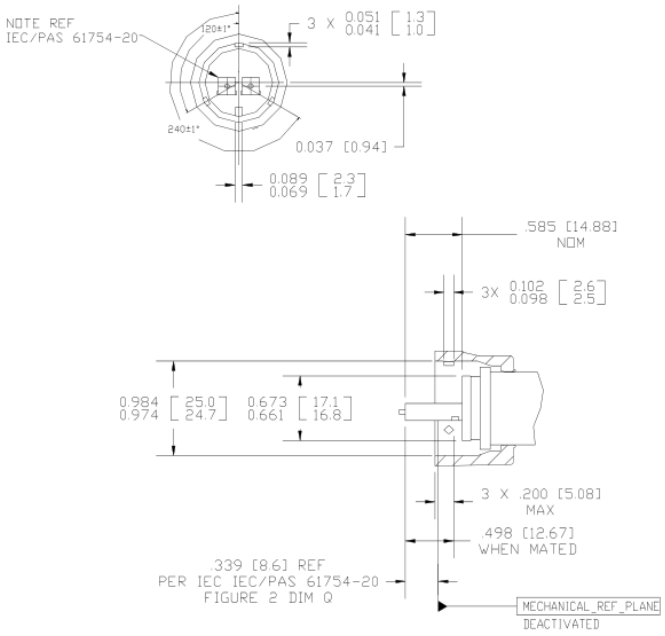
8-9.5.2.2 Sealed Industrial LC Connectors

Fiber optic connector designs shall meet the requirements of the corresponding ANSI/TIA/EIA (Fiber Optic Connector Intermateability Standard (FOCIS) documents). In the case where the LC fiber optic connector is placed into the IP65/67 shell or enclosure whereby the latch is defeated, the FOCIS requirements may not be applicable. See Note: IEC 61754-24 and EN 50377-6-1 is currently at CDV as of 10/10/08.

Table 8-9.21 LC, SC, SCRJ and ST Connector Insertion Loss.

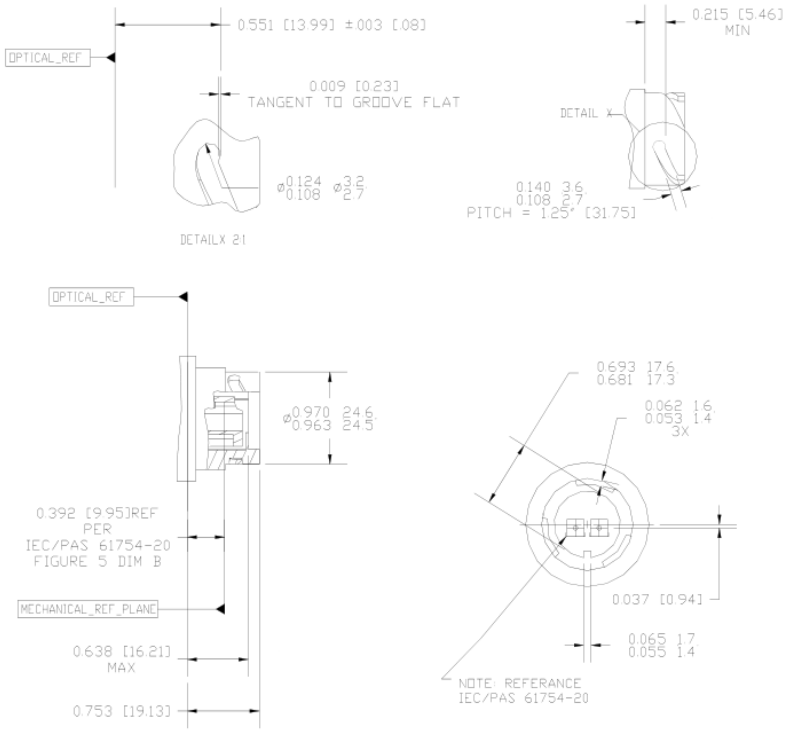
The following sealed plug drawing in Figure 8-9.15 defines the plug. The plug is fully compatible with standard off-the-shelf plugs with the exception of the defeated locking mechanism when placed in the Variant 1 housing. The dimensions are expressed in inches [mm].

Figure 8-9.15 Sealed Plug



The following sealed outlet/jack drawing in Figure 8-9.16 defines the outlet to maintain compatibility for mating and sealing amongst various vendors who make one or both parts. The outlet is fully compatible with off-the-shelf fiber optic LC plugs and jacks. The dimensions are expressed in inches [mm].

Figure 8-9.16 Sealed Outlet



8-9.5.2.3 Sealed M12 Fiber Optic Connector

The sealed M12 Fiber connector is RoHS[1] compliant with a small-form factor. See Table 8-9.22 for environmental, mechanical and optical characteristics.

Table 8-9.22 M12 Fiber Optic Connector Characteristics

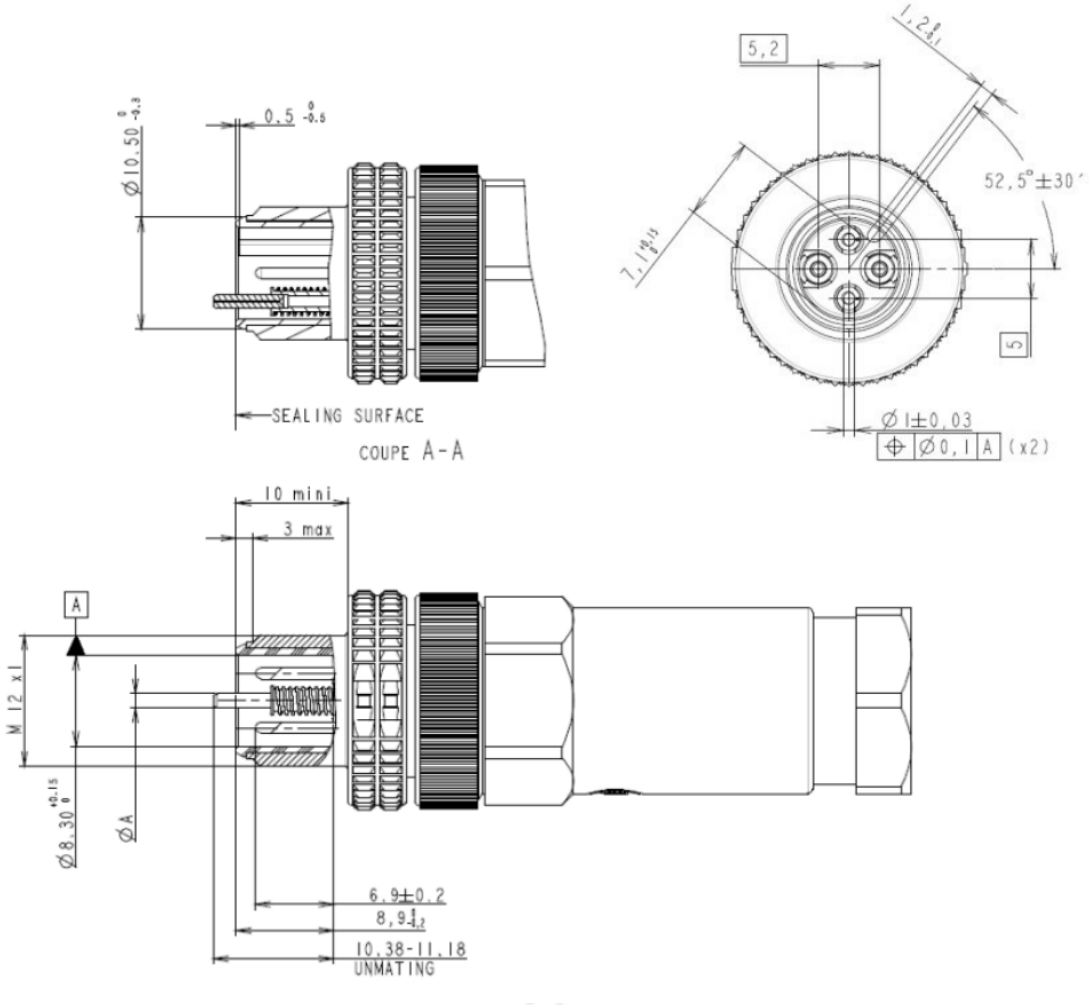
Characteristic	Range
Operating Temperature	-25C to +85C
Storage	-40C to +85C
Salt Spray	48 hours
Damp Heat	95% RH at 40C, per IEC 61076-2-101
IP67 rated M12 housing	Based on IEC-61076-2-101
Flame retardant	UL 94V-0
Durability	100 mating cycles
Contact Retention	20N
Cable Retention	80N Max
Cable Torsion	0.35 Nm +/- 30Deg
Typical Insertion loss	< 0.3 dB
Typical return loss	-30 dB
Multimode fibers 50/125 and 62.5/125	0.9 mm, 1.4 mm

[1] Restriction of Hazardous Substances directive.

The panel mount receptacle includes an integrated fiber optic transceiver, with a small form factor, package design and simplified assembly. See table 8-19 for transceiver characteristics.

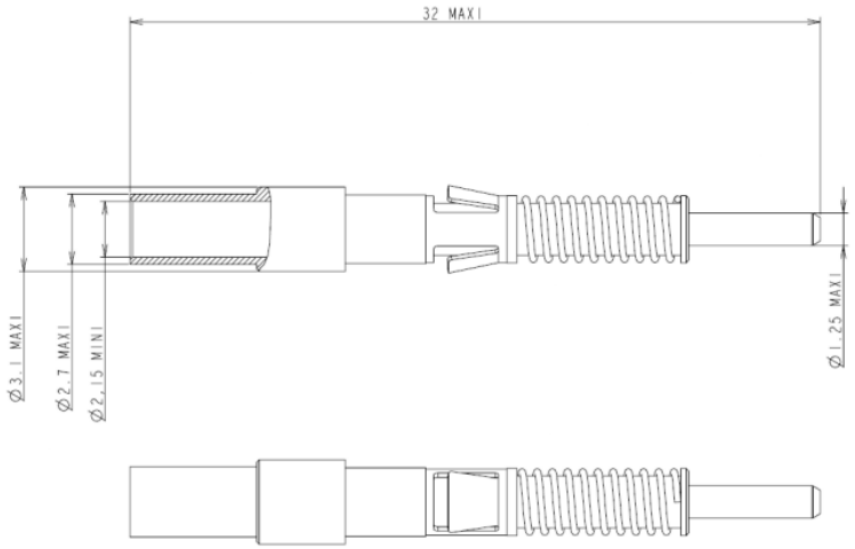
The sealed plug drawing in Figure 8-9.17 defines the plug. The cable plug includes a cable sealing strain relief and back shell, for enhanced sealing capability.

Figure 8-9.17 Connector Mechanical Dimensions (all dimensions in millimeters)



The connector shall be supplied with standard LC style 1.25mm ceramic ferrules in accordance with grade 2 of IEC 61754-20, as shown in Figure 8-9.18.

Figure 8-9.18 Ceramic ferrule details (all dimensions in millimeters)

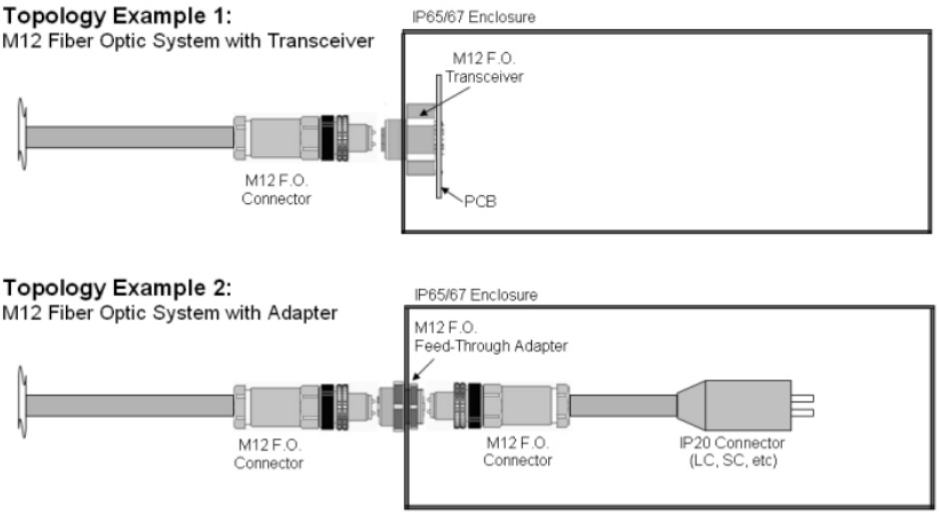


Optical Contact - Ceramic

8-9.5.2.4 Topology of Sealed M12 Fiber Connector

Figure 8-9.19 shows how the Sealed M12 Fiber Connector can be connected. The connector supports both end device connectivity to a like transceiver and in channel connectivity through a bulkhead connector.

Figure 8-9.19 Sealed M12 Fiber Connection Examples

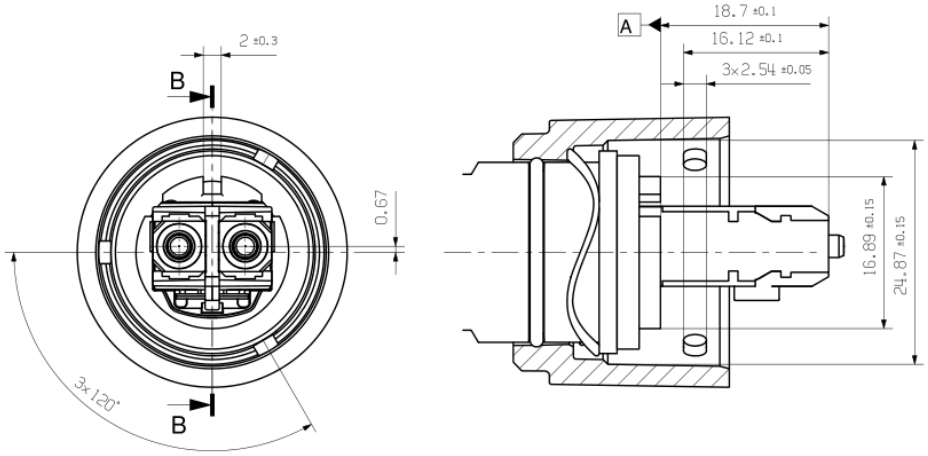


8-9.5.2.5 Sealed Industrial SCRJ Connector Plug

The SCRJ Fiber optic connector used in the IEC 61076-3-106 Variant 1 housing shall meet the requirements of IEC 61754-24. The plug shall be fully compatible with standard off-the-shelf plugs.

The following sealed plug drawing in Figure 8-9.20 defines the plug in the X, Y, and Z dimensions of the mating face of the Variant 1 plug housing with the SCRJ insert. All dimensions are expressed in mm. Dimensions are defined in the IEC 61076-3-106 _Variant 1.

Figure 8-9.20 Sealed SCRJ Plug



To ensure a proper mating and un-mating of the plug and jack assembly, the X,Y and Z dimensions shown in Figure 8-9.20 shall be used to locate the connector in the Variant 1 housing. In addition the solution covered in the Figure 8-9.21, Figure 8-9.22 and Figure 8-9.23 or a compatible solution should be used.

Note that the Variant 1 housing and the SCRJ connectors have independent locking mechanisms.

Figure 8-9.21 Mated Sealed SCRJ Plug, Fiber Deflected to Provide Spare Length

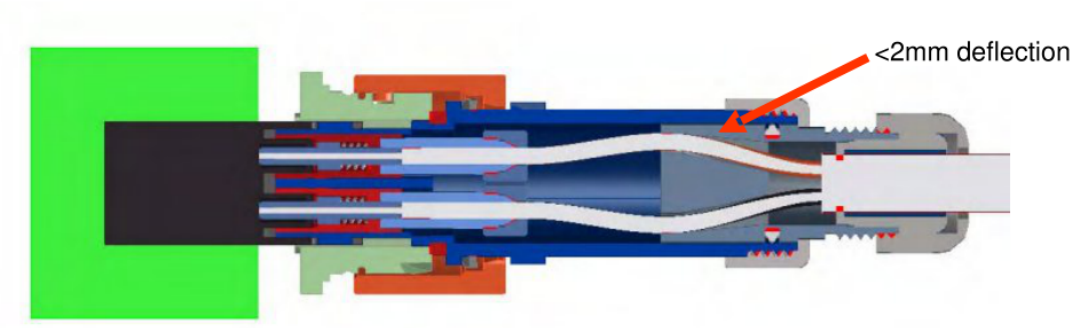
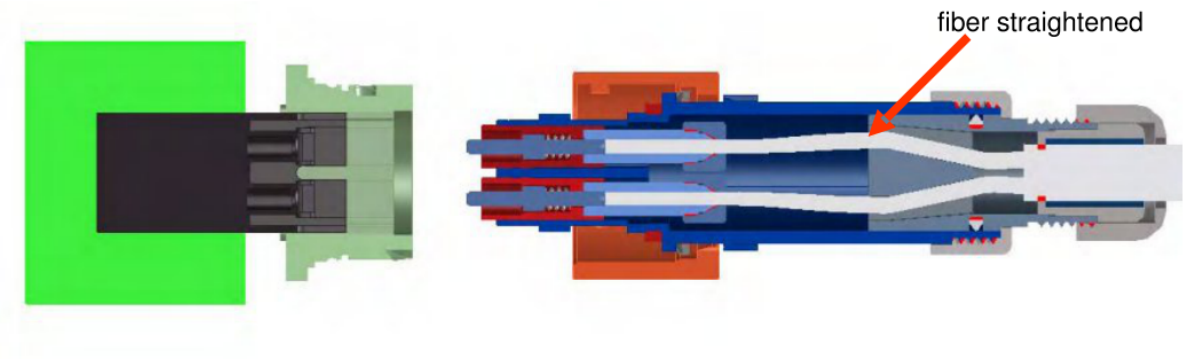
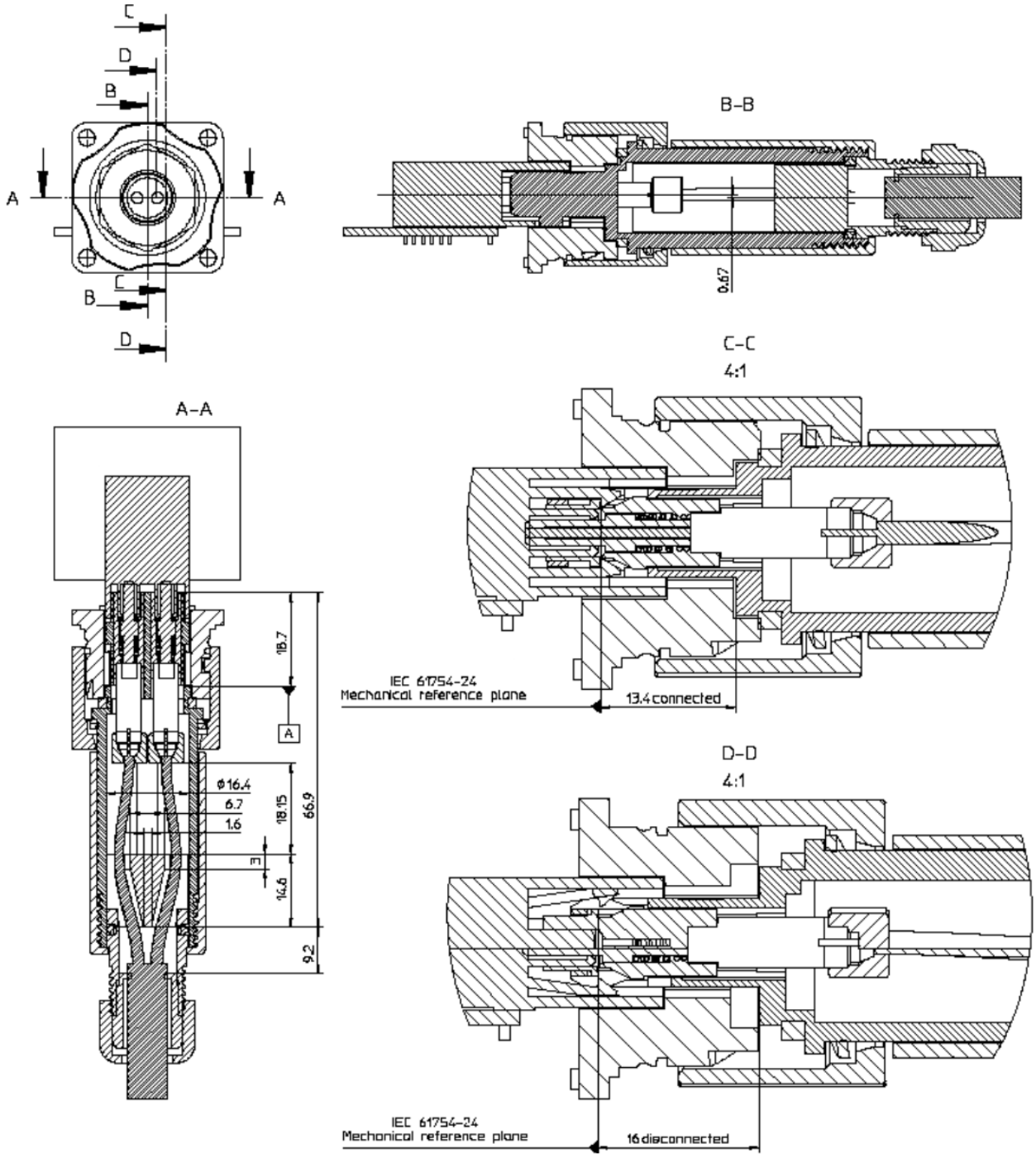


Figure 8-9.22 Unmated Sealed SCRJ Plug, Fiber Straightened to Unlock the SCRJ Insert



The sealed plug drawing in Figure 8-9.23 describes a design which provides the mechanism for the fiber deflection. The 2 mm fiber deflection accommodates disabling the latching mechanism of the IP 20 connection by allowing movement of the outer SCRJ latch housing. All dimensions are expressed in mm.

Figure 8-9.23 Example Design for the Fiber Deflection



Other latching and unlatching solutions that meet the mating compatibility requirements of the SCRG in the Variant 1 housing are also allowed for use in an ODVA compliant system.

The fiber termination technology for the POF fiber to the SCRJ ferrule is not part of this specification. Other commercially available termination technologies for POF are allowed and must meet the mating compatibility requirements of the SCRJ in the Variant 1 housing for use in an ODVA compliant system.

8-9.5.2.6 Sealed Receptacle

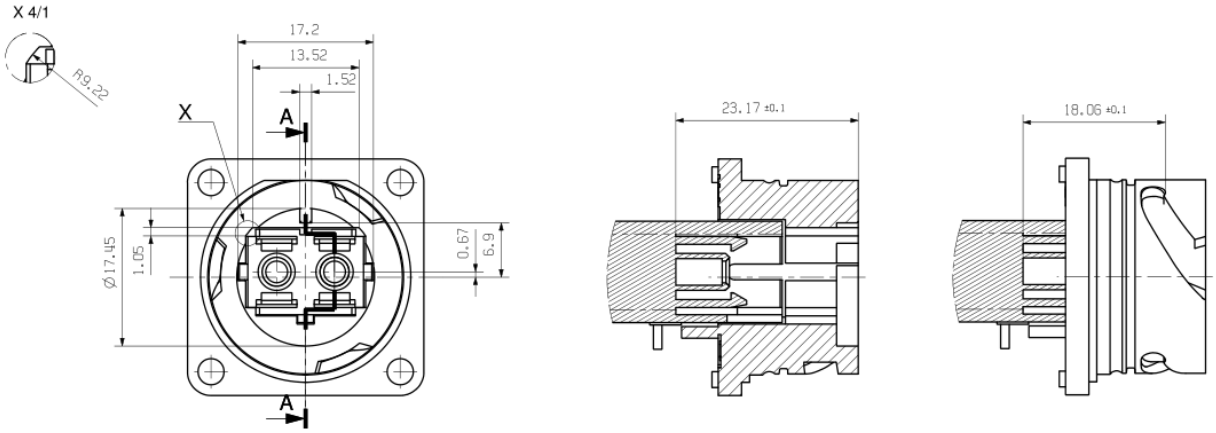
The following sealed receptacle drawing in Figure 8-9.24 defines the X, Y and Z dimensions of the Variant 1 receptacle with SCRJ coupler or transceiver. The SCRJ coupler or transceiver maintains full mating compatibility and sealing amongst various vendors who make one or both parts. The coupler or transceivers are fully compatible with standard off-the-shelf couplers or transceivers.

The receptacle is fully mating compatible with off-the-shelf fiber optic SCRJ plugs. It also accepts the SC plug connector. For testing, servicing and maintenance the latching mechanism is in use and the plug connector can be pulled out without tools. When the SCRJ fiber optic transceiver or coupler is installed in the IP65/67 Variant 1 receptacle, the following modification of the receptacle is required:

IEC 61076-3-106 Variant 1 Receptacle housing modification:

2 cuts in the receptacle housing (see Figure 8-9.24). Top left and top right of the receptacle housing as shown in the figure. All dimensions are expressed in mm. All other dimensions are as defined in the IEC 61076-3-106 Variant 1.

Figure 8-9.24 Sealed Receptacle Housing



8-9.5.3 Fiber PMD

The IEEE 802.3 standard defines many internal interfaces within the Physical Layer. EtherNet/IP products need not directly implement each of these interfaces, but shall behave "as if" these interfaces existed. These interfaces may be internal to the node and possibly internal to a semiconductor device. There are three media variants supported:

- 100BASE-LX10 using Single mode silica fibers;
- 100BASE-FX using multi mode silica fibers;
- 100 Mbps using Multi mode graded index plastic optical fiber, compatible with signaling of 100BASE-FX.

Other data rates are possible; however they are outside the scope of this standard and will not be compatible with the 100 Mbps fiber optic systems.

8-9.5.4 Fiber Optic Transceivers

This sub clause details the media specific transceivers for EtherNet/IP networks. Currently there are two media types supported, Single mode (SM) and Multimode (MM) fiber types. In addition this sub-clause provides details on the Sealed M12 transceiver that currently supports multimode fiber media. Future releases of this transceiver will include support for 1mm POF and single mode fiber media types. This transceiver is documented here to aid in the deployment of this new connector system.

8-9.5.4.1 Single mode

Fiber transceivers shall conform to IEEE 802.3 for 100BASE-LX10 (Ethernet in the First Mile) using SM Silica fibers. Additional requirements can be found in ISO/IEC 9314-3 Information processing systems-Fiber distributed Data Interface (FDDI)- part 3 Physical Layer Medium Dependant (PMD) standards with the exception of the transceiver which provides single mode coupling using the same wavelength of 1310nm as the multimode variant. The data rate shall be 100 Mbps.

The optical cabling power budget shall be a minimum of 10 dB.

Connectors supported (new designs):

- LC defined in 8-9.5.2.1
- Sealed LC defined in 8-9.5.2.2

The SC and ST connectors are allowable, however are not recommended for new designs.

8-9.5.4.2 Multimode

Fiber transceivers shall conform to IEEE 802.3 for 100BASE-FX when using multimode fibers. Additional requirements can be found in ISO/IEC 9314-3 Information processing systems-Fiber distributed Data Interface (FDDI)- part 3 Physical Layer Medium Dependant (PMD) standards.

The optical cabling power budget shall be a minimum of 11dB.

Connectors supported (new designs);

- LC defined in 8-9.5.2.1
- Sealed LC defined in 8-9.5.2.2

The SC and ST connectors are allowable, however are not recommended for new designs.

8-9.5.4.3 Transceiver for Sealed M12

The sealed M12 fiber optic transceiver detail is provided in this sub-clause.

Dimensional drawing for the Receptacle with the Integrated Transceiver is shown in Figure 8-9.25.

Figure 8-9.25 Transceiver Details (all dimensions in millimeters)

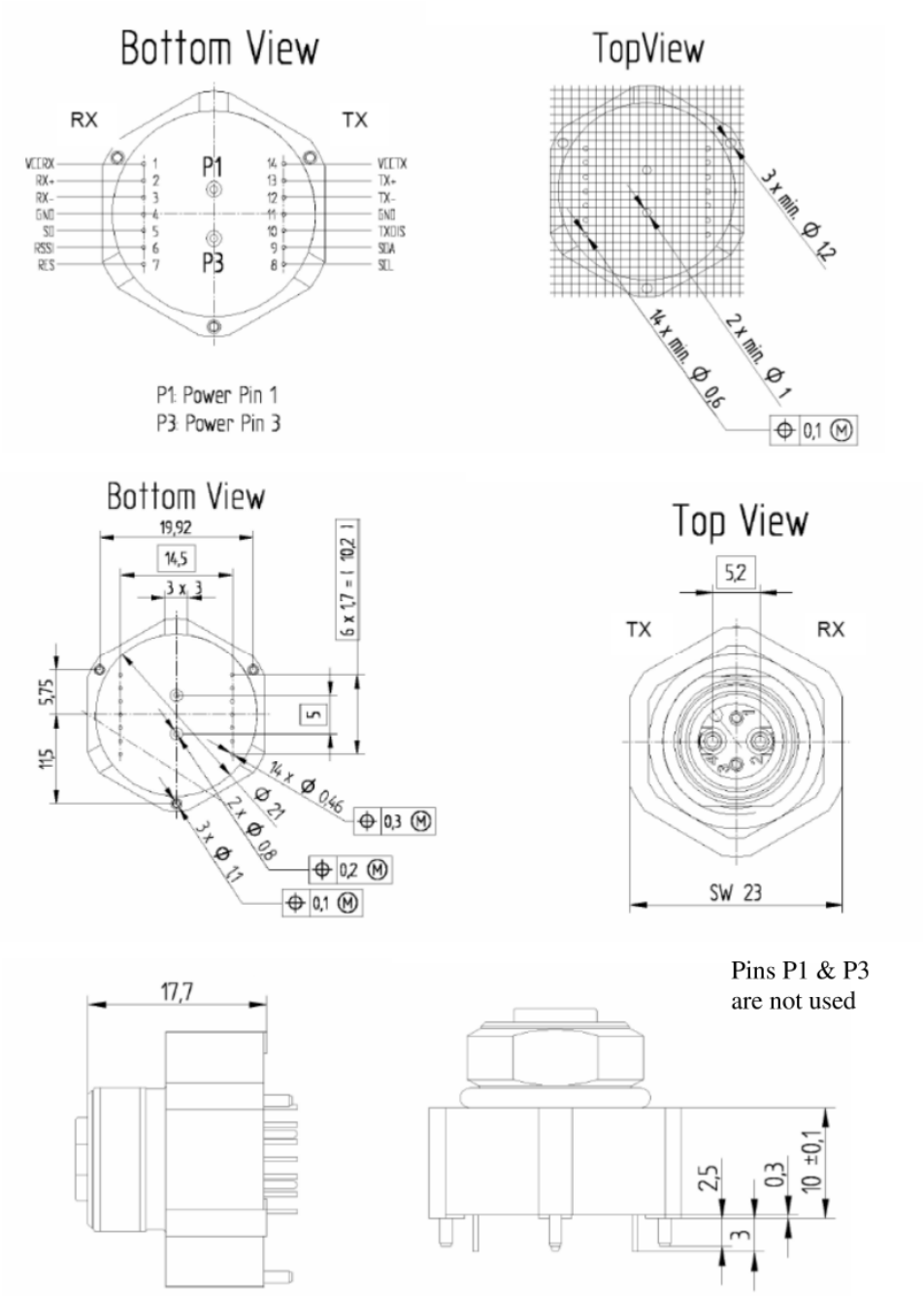


Table 8-9.23 Multimode transceiver characteristics

Characteristic	Value
Power supply	Single 3.3 V
Differential inputs and outputs	PECL / LVPECL
Sealing level	IP67 (mated)
Signal detect indicator	Yes
Center wave length:	1300 nm
Output optical power	Min 20 dBm to max -14 dBm coupled into 62.5um fiber
Input optical power range:	Min 14 dBm to Max -31 dBm

See Table 8-9.24 and Figure 8-9.25 for the transceiver pin out descriptions

Table 8-9.24 Pin Out Description

Pin Nr.	Symbol	Function
1	VCCRX	Receiver supply voltage, 3.3 Volt
2	RX+	Receiver data output, non-inverted, LVPECL
3	RX-	Receiver data output, inverted, LVPECL
4	GND	Ground (Receiver)
5	SD	Signal Detect, LVPECL
6	RSSI	Receive Signal Strength Indicator Output, analog voltage (optional)
7	RES	Reserved
8	SCL	Digital information interface, serial clock (optional)
9	SDA	Digital information interface, serial data (optional)
11	GND	Ground (Transmitter)
12	TX-	Transmitter data input, inverted, LVPECL
13	TX+	Transmitter data input, non-inverted, LVPECL
14	VCCTX	Transmitter supply voltage, 3,3 Volt
P1	P1	Pin for future use only. Do not connect to PCB trace.
P3	P3	Pin for future use only. Do not connect to PCB trace.

8-9.5.4.4 Step Index Multimode Transceivers

The POF transceivers shall conform to the electrical specifications of IEC 62149-6.

The PMD shall support to IEEE 802.3 for 100BASE-FX signaling when using step index multi mode fibers (e.g. POF). Additional requirements can be found in ISO/IEC 9314-3 Information processing systems-Fiber distributed Data Interface – Part 3 Physical Layer Medium Dependant (PMD) standards.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 9: Indicators & Middle Layers

Contents

9-1 Introduction

Chapter 9 specifies the standard appearance and behavior of EtherNet/IP diagnostic LEDs, basic TCP/IP requirements and defines the Device Level Ring protocol (DLR) for EtherNet/IP.

9-2 Data Link Layers

Though this specification is called “EtherNet/IP”, Ethernet is technically not required. The EtherNet/IP protocol may be used on any media that supports the transmission of the Internet Protocol.

NOTE: For example, the EtherNet/IP protocol could be used over FDDI, modem lines (SLIP or PPP), ATM, etc.

When any particular medium is used, it shall be used in accordance to commonly accepted standards. In particular, when Ethernet is used, it shall be used as defined by the IEEE 802.3 specification.

9-3 Requirements for TCP/IP Support

In addition to the various requirements set forth in this specification, all EtherNet/IP hosts are required to have a minimally functional TCP/IP protocol suite and transport mechanism. The minimum host requirements for EtherNet/IP hosts shall be those covered in RFC-1122, RFC-1123, and RFC-1127 and the subsequent documents that may supersede them. Whenever a feature or protocol is implemented by an EtherNet/IP host, that feature shall be implemented in accordance to the appropriate RFC documents, regardless of whether the feature or protocol is considered required or optional by this specification. The Internet and the RFCs are dynamic. There will be changes to the RFCs and to the requirements included in this section as the Internet and this specification evolves and these changes will not always provide for backward compatibility.

All EtherNet/IP devices shall at a minimum support:

- Internet Protocol (IP version 4) (RFC 791)
- User Datagram Protocol (UDP) (RFC 768)
- Transmission Control Protocol (TCP) (RFC 793)
- Address Resolution Protocol (ARP) (RFC 826)
- Internet Control Messaging Protocol (ICMP) (RFC 792)
- Internet Group Management Protocol (IGMP) (RFC 1112 & 2236)
- IEEE 802.3 (Ethernet) as defined in RFC 894

NOTE: Although the encapsulation protocol is suitable for use on other networks besides Ethernet that support TCP/IP and products may be implemented on these other networks, conformance testing of EtherNet/IP products is limited to those products on Ethernet. Other suitable networks include:

- Point to Point Protocol (PPP) (RFC 1171)
- ARCNET (RFC 1201)
- FDDI (RFC 1103)

NOTE: EtherNet/IP devices are encouraged but not required to support other Internet protocols and applications not specified here. For example, may support HTTP, Telnet, FTP, etc. This specification makes no requirements with regards to these protocols and applications.

9-4 Indicators

9-4.1 Required Indicators

A product need not have indicators to be compliant with this specification. However, to be compliant with the Industrial Performance Level described in Chapter 8, a product shall support both the module status and network status indicators as defined by sections 9-4.2, 9-4.3 and 9-4.4.

If a product does support any of the indicators described here, they must adhere to the specifications described in this section (section 9-4).

Two types of status indicators may be provided:

- One module status indicator;
- One network status indicator;

Additional indicators may be present; however, the naming and symbol conventions of the standard indicators shall not be employed for other indicators.

NOTE: Indicators, typically implemented as LEDs, help maintenance personnel to quickly identify a faulty unit or media. As such, red indicators are used to indicate a fault condition.

NOTE: Products are encouraged to have an indicator that displays the state of link (for example, link status, tx/rx, collision, etc.) following generally accepted industry practices (as used in devices such as switches).

9-4.2 Common Indicator Requirements

9-4.2.1 Applicability of Common Requirements

The common indicator requirement shall only apply to indicators for which requirements are specified in this standard.

9-4.2.2 Visibility of Indicators

Indicators shall be viewable without removing covers or parts from the equipment. Indicators shall be easily seen in normal lighting. Any labels and icons shall be visible whether or not the indicator is illuminated.

9-4.2.3 Indicator Flash Rate

Unless otherwise indicated, the flash rate of all indicators is approximately 1 flash per second. The indicator should be on for approximately 0.5 second and off for approximately 0.5 second. This flash rate specification only applies to the indicators specified in this chapter.

9-4.2.4 Indicators at Power Up

An indicator test is to be performed at power-up. To allow a visual inspection, the following sequence shall be performed:

- Turn first indicator Green, all other indicators off
- Leave first indicator on Green for approximately 0.25 second
- Turn first indicator on Red for approximately 0.25 second
- Turn first indicator on Green
- Turn second indicator (if present) on Green for approximately 0.25 second
- Turn second indicator (if present) on Red for approximately 0.25 second
- Turn second indicator (if present) Off

If other indicators are present, test each indicator in sequence as prescribed by the second indicator above. If a Module Status indicator is present, it shall be the first indicator in the sequence, followed by any Network Status indicators present. After completion of this power up test, the indicator(s) shall turn to a normal operational state.

9-4.3 Module Status Indicator

9-4.3.1 Description

The indication of module status shall require a single bicolor (red/green) indicator that represents the state of the entire product.

NOTE: A product with more than one communication port would have only one module status indicator, but more than one network status indicator (one per port).

9-4.3.2 Labeling

The module status indicator shall be labeled with one of the following:

- “MS”;
- "Mod";
- “Mod Status”;
- “Module Status”.

9-4.3.3 States

The module status indicator shall be in one of the following states:

Table 9-4.1 Module Status Indicator

Indicator state	Summary	Requirement
Steady Off	No power	If no power is supplied to the device, the module status indicator shall be steady off.
Steady Green	Device operational	If the device is operating correctly, the module status indicator shall be steady green.
Flashing Green	Standby	If the device has not been configured, the module status indicator shall be flashing green.
Flashing Red	Minor fault	If the device has detected a recoverable minor fault, the module status indicator shall be flashing red. **NOTE**: An incorrect or inconsistent configuration would be considered a minor fault.
Steady Red	Major fault	If the device has detected a non-recoverable major fault, the module status indicator shall be steady red.
Flashing Green / Red	Self-test	While the device is performing its power up testing, the module status indicator shall be flashing green / red.

9-4.4 Network Status Indicator

9-4.4.1 Description

The Network Status indicator shall be a bicolor (red/green) indicator that represents the status of the EtherNet/IP network interface. Devices with multiple physical communication ports but a single IP address shall have a single Network Status indicator (e.g., a 2-port device supporting Device Level Ring). Devices that support multiple IP addresses may use a Network Status indicator for each IP address interface, or may use a single indicator for all interfaces (per the behavior in Table 9-4.2).

Refer to Volume 2, Chapter 6 (Device Profiles) for additional information on devices with multiple interfaces.

9-4.4.2 Labeling

The network status indicator shall be labeled with one of the following:

- “NS”;
- “Net”;
- “Net Status”;
- "Network Status".

9-4.4.3 States

The network status indicator states shall be as follows:

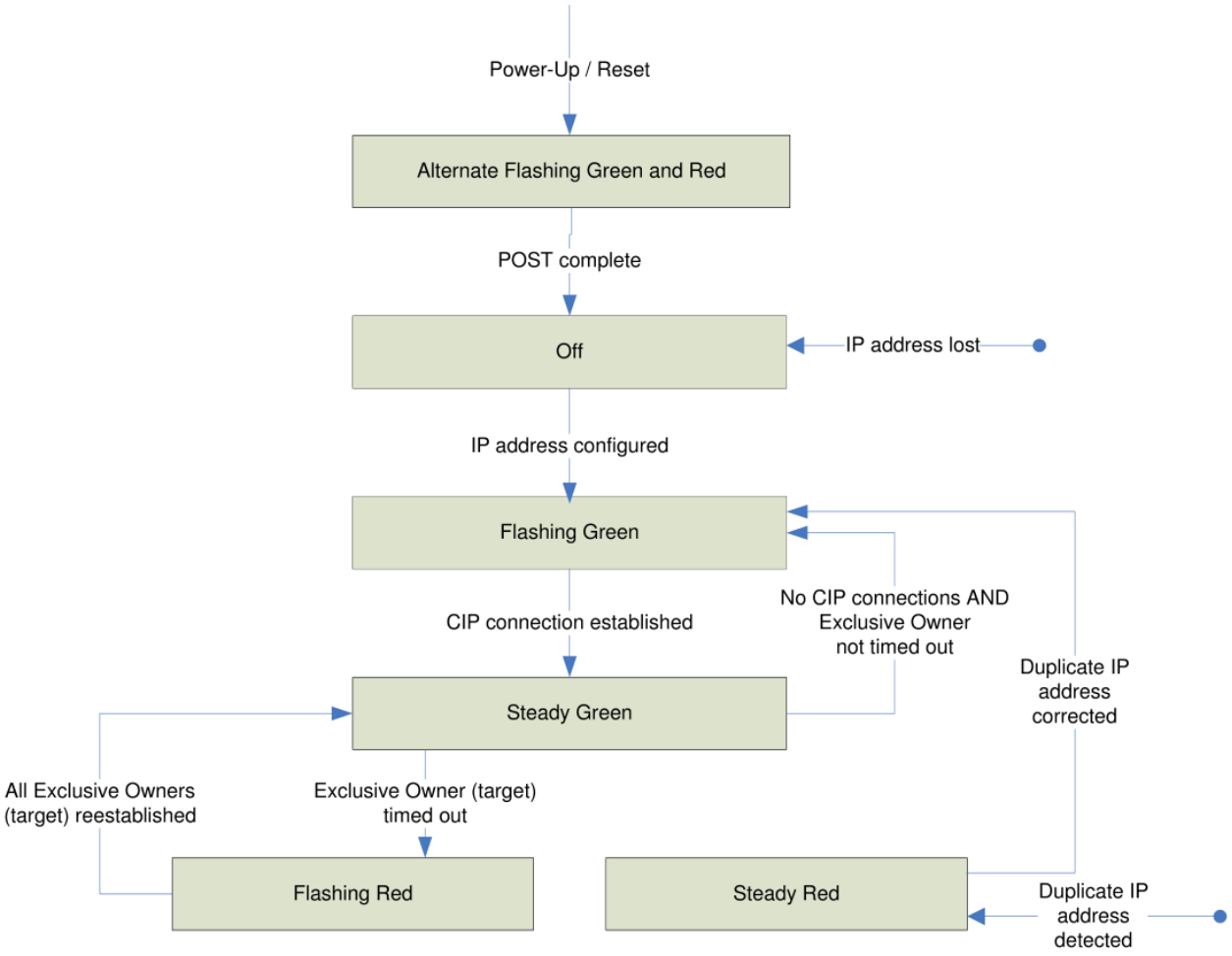
Table 9-4.2 Network Status Indicator

Indicator state	Summary	Requirement
Steady Off	Not powered, no IP address	The device is powered off, or is powered on but with no IP address configured (Interface Configuration attribute of the TCP/IP Interface Object).
Flashing Green	No connections	An IP address is configured, but no CIP connections are established, and an Exclusive Owner connection has not timed out.
Steady Green	Connected	At least one CIP connection (any transport class) is established, and an Exclusive Owner connection (defined in Volume 1, Chapter 3) has not timed out.
Flashing Red	Connection timeout	An Exclusive Owner connection (defined in Volume 1, Chapter 3) for which this device is the target has timed out. The network status indicator shall return to steady green only when all timed out Exclusive Owner connections are reestablished. Devices that support a single Exclusive Owner connection shall transition to steady green when any subsequent Exclusive Owner connection is established. Devices that support multiple Exclusive Owner connections shall retain the O->T connection path information when an Exclusive Owner connection times out. The Network LED shall transition from flashing red to steady green only when all connections to the previously timed-out O->T connection points are reestablished. Timeout of connections other than Exclusive Owner connections shall not cause the indicator to flash red. The Flashing Red state applies to target connections only. Originators and CIP Routers shall not enter this state when an originated or routed CIP connection times out.
Steady Red	Duplicate IP	For devices that support duplicate IP address detection, the device has detected that (at least one of) its IP address is already in use.
Flashing Green and Red	Self-test	The device is performing its power-on self-test (POST). During POST the network status indicator shall alternate flashing green and red.

Note: when a single indicator is used to represent multiple IP address interfaces the state of any one interface shall be sufficient to modify the indicator state (per the above behavior in the table):

- Transition to flashing green when any one interface receives an IP address
- Transition to steady green when a CIP connection is established on any interface (and Exclusive Owner is not timed out).
- Transition to flashing red when an Exclusive Owner CIP connection times out on any interface
- Transition to steady red when any of the interfaces detects an IP address conflict

Figure 9-4.1 Network Status Indicator State Diagram



9-5 Device Level Ring Protocol

9-5.1 Introduction

This section defines the Device Level Ring (DLR) protocol – a Layer 2 protocol that provides media redundancy in a ring topology. The DLR protocol is intended primarily for implementation in EtherNet/IP end devices that have multiple Ethernet ports and embedded switch technology. The DLR protocol provides for fast network fault detection and reconfiguration in order to support the most demanding control applications.

Since the DLR protocol operates at Layer 2 (in the OSI network model), the presence of the ring topology and the operation of the DLR protocol are transparent to higher layer protocols such as TCP/IP and EtherNet/IP, with the exception of a DLR Object that provides a configuration and diagnostic interface for EtherNet/IP.

A DLR network includes at least one node configured to be a ring supervisor, and any number of normal ring nodes. It is assumed that all the ring nodes have at least two Ethernet ports and incorporate embedded switch technology.

Non-DLR multi-port devices – switches or end devices – may be placed in the ring, subject to certain implementation constraints (e.g., no MAC table filtering). Non-DLR devices will also impact the worst-case ring recovery time.

The DLR protocol definition includes a number of aspects:

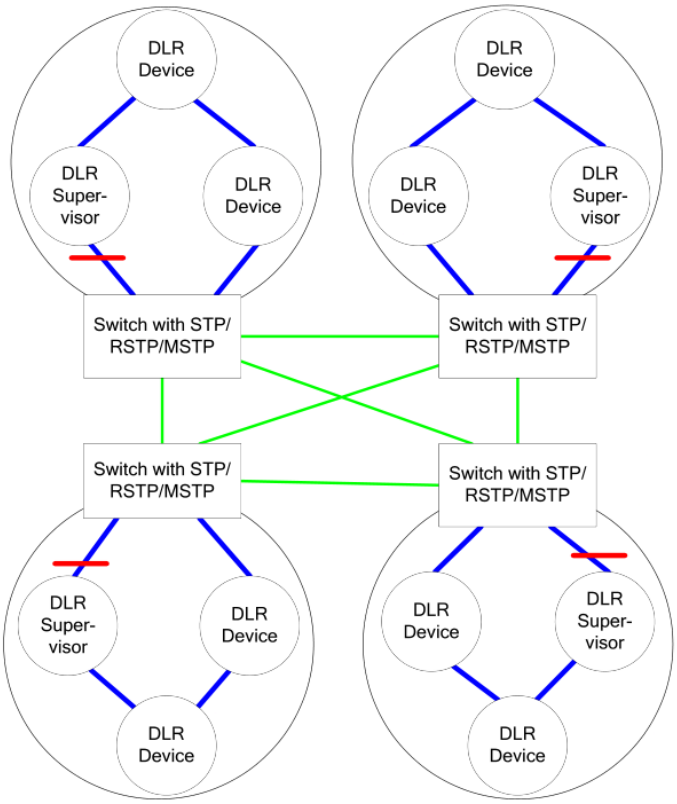
- A set of end node behaviors, for ring supervisors and normal ring nodes.
- Protocol messages and associated state diagrams
- Implementation requirements for devices

9-5.2 Supported Topologies

The DLR protocol supports a simple, single-ring topology; it has no concept of multiple or overlapping rings. A network installation may however use more than one DLR-based ring, so long as each of the rings are isolated such that DLR protocol messages from one ring are not present on another ring.

The DLR protocol may coexist with, but does not interface with, standard network protocols such as IEEE Spanning Tree Protocols (STP, RSTP, MSTP), and also with vendor-specific redundancy protocols. That is, users may construct network topologies with DLR protocol rings connected to switches that are running Spanning Tree or other ring protocols, as shown in Figure 9-5.1.

Figure 9-5.1 DLR Rings Connected to Switches



In Figure 9-5.1, each DLR ring is a separate DLR network, each with a ring supervisor. The supervisors are shown with one port in blocked mode, which is the case when there are no faults in the ring.

The switches to which the DLR rings are connected may run STP/RSTP/MSTP to ensure loop free operation when redundant paths are present (indicated by the green lines in Figure 9-5.1). Spanning Tree Protocol messages (BPDUs) that are sent by the switch on the DLR ring ports will be blocked by the DLR Ring Supervisor, so that the switches do not block the DLR ports (refer to Section 9-5.6.4, IEEE 802.1D/802.1Q STP/RSTP/MSTP Considerations).

Note: The switches' ports to which the DLR devices are connected must be configured properly in order ensure proper functioning of the network (refer to section 9-5.5).

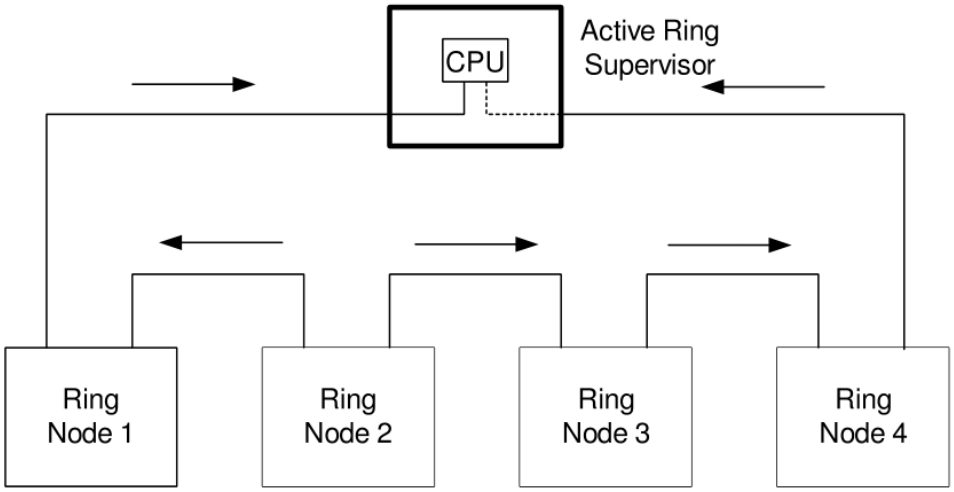
More complicated topologies combining DLR rings and non-DLR switches running STP/RSTP/MSTP may result in DLR ports being blocked in an undesirable manner. Refer to section 9-5.5 for additional information.

9-5.3 Overview of DLR operation

9-5.3.1 Normal Operation

Figure 9-5.2 shows the normal operation of a DLR network.

Figure 9-5.2 Normal DLR Network Operation

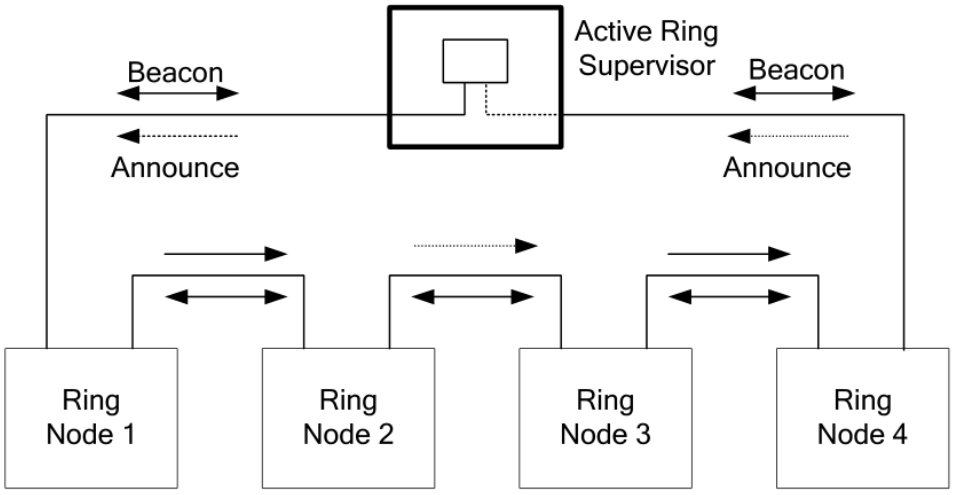


Each node in Figure 9-5.2 has two Ethernet ports, and is assumed to have implemented an embedded switch. When a ring node receives a packet on one of its Ethernet ports, it determines whether the packet needs to be received by the ring node itself (e.g., the packet has the node's MAC address) or whether the packet should be sent out the node's other Ethernet port.

The active ring supervisor blocks traffic on one of its ports with the exception of few special frames and does not forward traffic from one port to other. Because of this configuration a network loop is avoided and only one path exists between any two ring nodes during normal operation.

Figure 9-5.3 illustrates the use of Beacon and Announce frames sent by the active ring supervisor:

Figure 9-5.3 Beacon and Announce frames



The active ring supervisor transmits a Beacon frame through both of its Ethernet ports once per beacon interval (400 microseconds by default). The supervisor also sends Announce frames once per second. The Beacon and Announce frames serve several purposes:

1. The presence of Beacon and Announce frames inform ring nodes to transition from linear topology mode to ring topology mode.

2. Loss of Beacon frames at the supervisor enables detection of certain types of ring faults. (Note that normal ring nodes are also able to detect and signal ring faults).

3. The Beacon frames carry a precedence value, allowing selection of an active supervisor when multiple ring supervisors are configured.

9-5.3.2 Link Failures

9-5.3.2.1 Common Failures

The most common form of link failure includes the following cases:

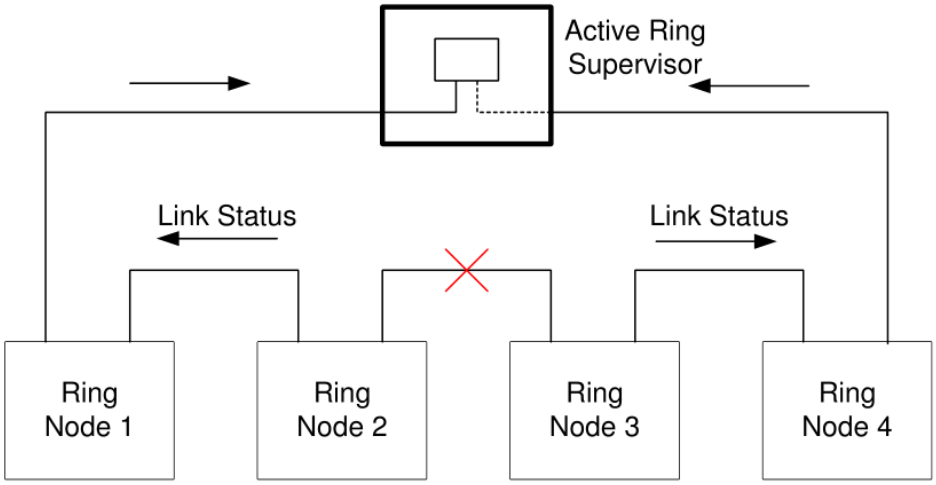
1. Link or other physical layer failure recognized by a node adjacent to the failure.

2. Power failure or power cycling a ring node, recognized by the adjacent node as a link failure.

3. Intentional media disconnect by user to bring new nodes online or to remove existing ones.

In the above cases, the nodes adjacent to the fault send a Link_Status message to the ring supervisor. Figure 9-5.4 shows ring nodes adjacent to a fault sending a Link_Status message to the ring supervisor.

Figure 9-5.4 Link Failure

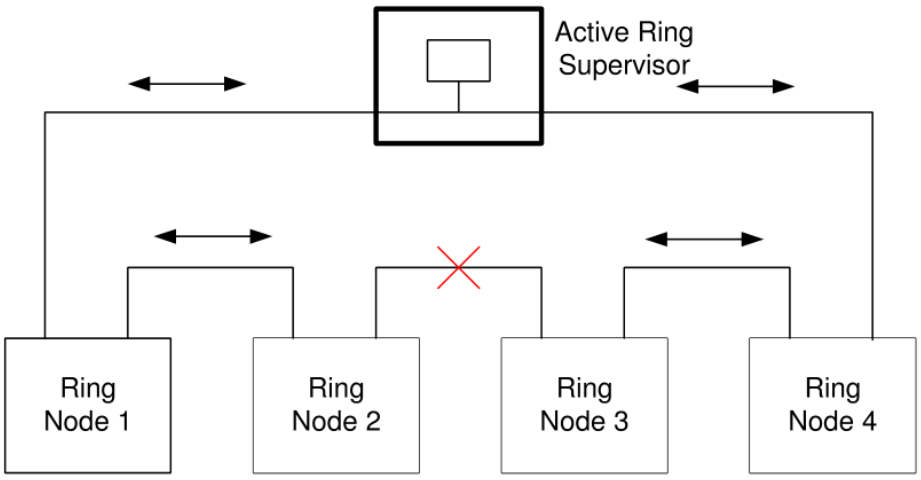


After receipt of the Link_Status message, the ring supervisor reconfigures the network by unblocking traffic on its previously blocked port and flushing its unicast MAC table. The supervisor immediately sends Beacon and Announce frames with the ring state value indicating that the ring is now faulted.

Ring nodes also flush their unicast MAC tables upon detecting loss of the beacon in one direction, or upon receipt of Beacon or Announce frames with the ring state value indicating the ring fault state. Flushing the unicast MAC tables at both supervisor and ring nodes is necessary for network traffic to reach its intended destination after the network reconfiguration.

Figure 9-5.5 shows the network configuration after a link failure, with the ring supervisor passing traffic through both of its ports.

Figure 9-5.5 Network Reconfiguration after Link Failure



9-5.3.2.2 Uncommon Failures

In addition to the more common link failures, there is a class of uncommon failures:

1. Higher level hardware/firmware component(s) on a ring node has failed leading to lost traffic, but the physical layer is functioning normally with power supply intact.

2. A chain of ring protocol unaware nodes are connected between protocol-aware nodes, and the failure has occurred somewhere in the middle of this chain.

In these cases, the ring supervisor will detect the loss of Beacon frames first on one port, and eventually on both of its ports. The supervisor will reconfigure the network as described in the "Common Failures" section. In addition, the ring supervisor will send a Locate_Fault frame to diagnose the fault location (refer to the subsequent section on the Neighbor Check process).

9-5.3.2.3 Partial Network Fault

It is possible for a partial network fault to occur such that traffic is lost in only one direction. The active ring supervisor detects a partial fault by monitoring the loss of Beacon frames on one port. When a partial fault is detected the active supervisor blocks traffic on one port and sets a status value in the DLR Object. The ring at this point will be segmented due to the partial fault, requiring user intervention.

9-5.3.2.4 Rapid Fault/Restore Cycles

Certain conditions such as a faulty network connector may cause the ring supervisor to detect a series of rapid fault/restore cycles. If left to persist, such a condition could result in network instability that is difficult to diagnose. When the active supervisor detects the rapid fault/restore condition (5 faults in a 30 second period), it sets a status value in the DLR Object, and blocks traffic on one port. The user must explicitly clear the condition via the DLR Object.

9-5.4 Classes of DLR Implementation

There are several classes of DLR implementation, as described below. Detailed requirements for each class of implementation are further specified in subsequent sections.

9-5.4.1 Ring Supervisor

This class of device is capable of being a ring supervisor. Such devices must implement the required ring supervisor behaviors, including the ability to send and process Beacon frames at the default beacon interval of 400 microseconds. Smaller beacon intervals, as low as 100 microseconds, may be supported, but are not required.

9-5.4.2 Ring Node, Beacon-based

This class of device implements the DLR protocol, but without the ring supervisor capability. The device must be able to process and act on the Beacon frames sent by the ring supervisor. Beacon-based ring nodes shall support beacon rates of 100 microseconds to 100 milliseconds in order to accommodate all ring supervisor implementations.

9-5.4.3 Ring Node, Announce-based

This class of device implements the DLR protocol, but without the ring supervisor capability. In order to accommodate nodes that do not have the capacity to process Beacon frames, ring nodes may simply forward, but not explicitly process, Beacon frames. Such nodes must process Announce frames.

9-5.5 DLR Behavior

9-5.5.1 DLR Variables

Table 9-5.1 summarizes variables used in the DLR protocol behavior and messages. Refer to the subsequent sections on Ring Node and Ring Supervisor behavior and DLR messages for further details. The DLR Object (Chapter 5) exposes these variables (with the exception of the Node State) via object attributes.

Table 9-5.1 DLR Variables

DLR Variable	Description
Node State	Internal state of a node's DLR state machine: IDLE_STATE – initial state for non-supervisors, indicating linear topology mode. FAULT_STATE – initial state for enabled ring supervisor, or when ring fault has been detected (both supervisor and ring nodes). NORMAL_STATE – normal function in ring topology mode.
Ring State	State of the ring network. Transmitted by ring supervisors in Beacon and Announce frames: RING_NORMAL_STATE – Ring is functioning, with supervisor blocking traffic on one port. RING_FAULT_STATE – Fault detected, ring supervisor is not blocking traffic (also is the initial state transmitted in the Beacon and Announce frames).
Beacon Interval	Interval at which the ring supervisor sends Beacon frames. Supervisors shall support a range from 400 microseconds to 100 milliseconds. The default value shall be 400 microseconds. Supervisors may support a Beacon Interval smaller than 400 microseconds, but this is not required. The absolute minimum Beacon Interval is 100 microseconds. Beacon-based ring nodes shall support beacon rates of 100 microseconds to 100 milliseconds in order to accommodate all ring supervisor implementations.
Beacon Timeout	Amount of time nodes shall wait before timing out reception of Beacon frames and taking the appropriate action (depending on whether supervisor or normal ring node). Supervisors shall support a range from 800 microseconds to 500 milliseconds. The default shall be 1960 microseconds. Supervisors may support a Beacon Timeout of smaller than 800 microseconds but this is not required. The absolute minimum Beacon Timeout is 200 microseconds.
Supervisor Precedence	Precedence value assigned to a ring supervisor, and transmitted in Beacon frames. Used to select active ring supervisor when multiple supervisors have been configured. Default value is 0. Can be changed via the DLR Object.
DLR VLAN ID	VLAN ID used when sending DLR protocol frames. The VLAN ID is configured at the ring supervisor (via the DLR Object), and is then detected by ring nodes when they receive and process the Beacon or Announce frames from the supervisor. Default value is 0. Typically the VLAN ID does not need to be changed unless a commercial switch is being used in the ring.

9-5.5.2 Ring Supervisor

9-5.5.2.1 Startup

An enabled ring supervisor shall start in FAULT_STATE and configure both ports to forward frames. The supervisor shall send Beacon frames out both of its ports, with the Ring State set to RING_FAULT_STATE. The supervisor shall also send Announce frames out both of its ports with the Ring State set to RING_FAULT_STATE.

Once the Beacon frames are received through both ports the supervisor shall transition to NORMAL_STATE, flush its unicast MAC address table and reconfigure one of its ports not to forward packets, except for the following, which shall be forwarded to the host for processing:

- Beacon frames with the supervisor's own MAC address (in general needed only for software implementations).
- Beacon frames from other ring supervisors.
- Link_Status/Neighbor_Status frames.
- Neighbor_Check request or response, and Sign_On: always forward received frames. For frames originated by the supervisor, only forward frames with the Source Port matching the blocked port.

Upon transition to NORMAL_STATE, the Ring State in the Beacon frames shall be set to RING_NORMAL_STATE. The ring supervisor shall also send an Announce frame out one port, with Ring State set to RING_NORMAL_STATE.

9-5.5.2.2 Multiple Ring Supervisors

When multiple ring supervisors are configured, each supervisor sends Beacon frames when it comes online. The Beacon frames carry a supervisor precedence value. When a supervisor receives a Beacon frame, it checks the precedence value. If the precedence in the Beacon frame is higher than the receiving node's precedence value, the receiving node transitions to FAULT_STATE and becomes a backup supervisor. If the precedence values are the same, the node with the numerically higher MAC address becomes the active supervisor.

The backup supervisors configure their DLR parameters with the values obtained from the active supervisor's Beacon frames: Beacon Interval, Beacon Timeout, VLAN ID.

The backup supervisors continue to monitor both ports for timeout of the Beacon frames (no Beacons received within the Beacon Timeout period). If the Beacon has timed out on both ports, the backup supervisor waits for an additional Beacon Timeout period (during which time other nodes transition to linear mode), then begins sending its own Beacons so that a new supervisor can be selected.

9-5.5.2.3 Sign On

In order to identify ring protocol participants, the active ring supervisor shall send a Sign_On frame when it transitions to NORMAL_STATE. Refer to the Sign_On information in the "DLR Messages" section.

9-5.5.2.4 Normal Ring Operation

When in the NORMAL_STATE, the active ring supervisor shall send Beacon frames out both of its ports. It shall also send an Announce frame once per second out one port

One of the active supervisor's ports shall be configured not to forward frames, with the exceptions as noted in section 9-5.5.2.1.

9-5.5.2.5 Ring Fault Detection

One of several possible events shall cause the active ring supervisor to transition to FAULT_STATE:

1. Beacon frame received from another supervisor with a higher precedence value.

2. Loss of Beacon frames on either port for the period specified by the Beacon Timeout, indicating a break somewhere in the ring.

3. Detection of loss of link with the neighboring node on either port.

4. Link_Status frame received from a ring node, indicating a ring node has detected a fault.

In all of the cases listed above, the active ring supervisor shall:

- Transition to FAULT_STATE
- Flush its unicast MAC address table
- Unblock the blocked port
- Send Beacon frame out both ports, with Ring State set to RING_FAULT_STATE
- Send Announce frame out both ports, with Ring State set to RING_FAULT_STATE

In addition, in case 2 above, the active ring supervisor shall initiate the Neighbor Check process by issuing a Locate Fault frame. The supervisor shall also issue its own Neighbor Check frame through the port(s) on which the beacon has timed out.

When in FAULT_STATE the ring supervisor shall continue to send Beacon frames, in order to detect ring restoration.

9-5.5.2.6 Ring Restoration

When the active ring supervisor is in FAULT_STATE, receipt of Beacon frames on both ports shall cause a transition to NORMAL_STATE. The active ring supervisor shall do the following:

- Flush the unicast MAC address table
- Reconfigure its ports such that traffic is not forwarded on one port (with exceptions as noted previously)
- Send Beacon frames with the Ring State set to RING_NORMAL_STATE
- Send Announce frames out one port with Ring State set to RING_NORMAL_STATE

9-5.5.2.7 Changing Ring Parameters

The following ring supervisor parameters may be changed via the DLR Object (see Chapter 5):

- Supervisor precedence value
- Beacon interval
- Beacon timeout
- VLAN ID
- Supervisor enabled/disabled

When any of the above parameters are changed on the active ring supervisor, the ring supervisor shall cease sending Beacon and Announce frames for two (current) beacon timeout periods, then shall send Beacon frames using the new parameters. Ceasing the Beacon frames allows beacon-based ring nodes to detect the new parameters when the Beacon is restored and triggers active supervisor negotiation based on the new parameters. Announce frame production is further suppressed for at least 2 new beacon timeout periods to make sure that the active supervisor is determined before any Announce frames with the new parameters are sent.

When parameters are changed on a backup ring supervisor, the behavior depends on the backup supervisor's new precedence value compared to the active supervisor's precedence value:

- New backup precedence value is greater than the current active ring supervisor's precedence or of equal precedence with numerically higher MAC address than active supervisor MAC address: backup shall immediately begin sending Beacon frames with the new parameters.
- New backup precedence value is less than the active supervisor's precedence or of equal precedence with numerically lower MAC address than active supervisor MAC address: modification to the Beacon Interval, Beacon Timeout, and VLAN ID shall be ignored.

9-5.5.3 Ring Node

9-5.5.3.1 Beacon VS. Announce-Based Implementations

Ring nodes (that is, non-supervisor nodes) may have differing implementations depending on whether or not they are able to process the Beacon frames which by default are sent every 400 microseconds (but may be sent as fast as every 100 microseconds). Nodes that are able to process the Beacon frames generally have hardware assistance in implementing the DLR protocol, so that they don't burden the device's CPU with processing the Beacon frames.

Devices that would need to process the Beacon frames in the device's CPU can instead configure their embedded switch to simply pass the Beacon frames on the network without interpretation or further processing. Such devices must however process the Announce frames, which also indicate the ring state but are sent at a much slower rate.

Note that it is possible to implement a Beacon-based node without hardware assistance, provided the device's CPU has sufficient capacity to process the Beacon frames in addition to its other required functions.

It is desirable for device implementations to be Beacon-based rather than Announce-based, since better ring recovery performance results when ring nodes are able to process Beacon frames. Refer to section 9-5.10 (Performance Analysis).

9-5.5.3.2 Startup – Beacon-based

A Beacon-based ring node shall start up in IDLE_STATE, which presumes the network is in linear topology mode.

Upon receiving a Beacon frame through either port, the node shall transition to FAULT_STATE, which presumes the ring topology mode. The ring node shall flush its unicast MAC address table and save the ring supervisor parameters from the Beacon frame:

- Supervisor MAC address
- Supervisor precedence value
- Beacon timeout
- VLAN ID

Upon receiving Beacon frames through both ports and after receiving a Beacon frame from active ring supervisor with ring state field set to RING_STATE_NORMAL on either one of its ports, the node shall transition to NORMAL_STATE and flush its unicast MAC address table.

9-5.5.3.3 Startup – Announce-based

An Announce-based ring node shall start up in IDLE_STATE, which presumes the network is in linear topology mode.

Upon receiving an Announce frame through either port, the node shall transition to the ring state indicated in the Announce frame. The ring node shall flush its unicast MAC address table and save the ring supervisor parameters from the Announce frame:

- Supervisor MAC address
- Ring State
- VLAN ID

9-5.5.3.4 FAULT DETECTION

One of several possible events shall cause a ring node to transition from NORMAL_STATE:

For Beacon-based nodes:

1. Receipt of a Beacon frame with the Ring State set to RING_FAULT_STATE.

2. Receipt of a Beacon frame with a different MAC address and higher precedence than the current ring supervisor.

3. Loss of Beacon frames on both ports for the period specified by the Beacon Timeout, which causes the node to transition to IDLE_STATE (i.e., the topology is now linear).

4. Loss of Beacon on a single port for the period specified by the Beacon Timeout.

For Announce-based nodes:

1. Receipt of an Announce frame with the Ring State set to RING_FAULT_STATE

2. Loss of Announce frame for the Announce timeout duration, which causes the node to transition to IDLE_STATE (i.e., the topology is now linear).

In all of the cases listed above, the ring node shall:

- Flush its unicast MAC address table.
- Transition to FAULT_STATE (Exception: loss of Beacon on both ports or loss of Announce causes transition to IDLE_STATE).

9-5.5.3.5 Ring Restoration

For ring nodes, the process for ring restoration is the same as the Startup case (see section 9-5.5.3.2 and 9-5.5.3.3).

9-5.5.4 Sign On Process

The Sign_On frame is used to identify all ring participants. The active ring supervisor shall send a Sign_On frame when it transitions to NORMAL_STATE. The active supervisor transmits a Sign_On frame once every one minute while in NORMAL_STATE, until it receives a Sign_On that it sent out previously. Upon receiving such a frame the active supervisor will cease to send further Sign_On frames until next transition into NORMAL_STATE. The collected participant list can be accessed through the DLR Object.

The Sign_On frame is a multicast message transmitted from one port of the active ring supervisor. The receiving ring participant node traps the Sign_On frame and forwards it only to the host CPU. The host CPU increments the number of nodes in list, add its own addresses to list and transmit the Sign_On frame only through the other port than the receiving port.

The Sign_On frame is transmitted from one ring participant node to the next in similar fashion and eventually reaches the active supervisor. The active supervisor can identify the Sign_On frame it sent out by confirming that the first entry is its own.

It is possible that the number of nodes in the ring large enough that all nodes' addresses do not fit in the Sign_On frame. When a node receives the Sign_On frame, if adding the node's address would exceed the maximum frame size, the node does not add its address, but saves the port through which the frame was received and sends the Sign_On frame directly to the active ring supervisor.

When the ring supervisor receives the Sign_On frame sent to its unicast MAC address, it assumes this is due to the Sign_On frame size reaching its maximum. The supervisor restarts the Sign On process by sending a new Sign_On frame directly (unicast) to the node from which it received the unicast Sign_On frame.

Upon receiving the new Sign_On frame from the ring supervisor, the ring node adds its address to the Sign_On frame. The node then sends the Sign_On frame (multicast) through its other port than the saved port.

9-5.5.5 Neighbor Check Process

When the active ring supervisor detects the loss of Beacon, it sends a Locate_Fault frame through both ports.

Upon receipt of the Locate_Fault frame, each ring node issues a Neighbor_Check request through both of its ports. The supervisor also issues its own Neighbor_Check request.

When any node receives a Neighbor_Check_Request frame it responds with a Neighbor_Check_Response frame through the port on which original request was received. If the node sending the Neighbor_Check_Request does not receive a response in 100ms, it shall retry the request. After a total of 3 attempts, if no response is received after the overall 300ms timeout expires, the node sends a Neighbor_Status frame to the ring supervisor.

Figure 9-5.6 Neighbor Check Process

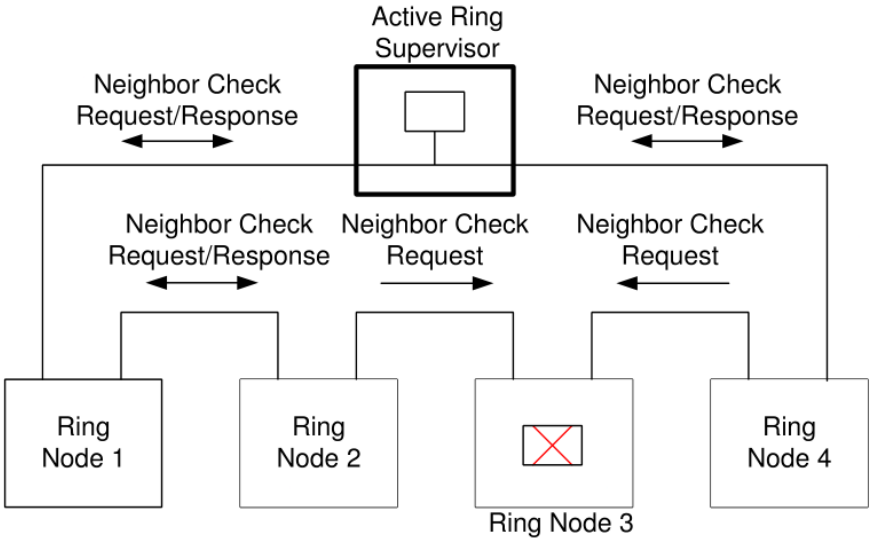


Figure 9-5.6 illustrates the Neighbor Check process. In this example, all healthy ring nodes respond, while the failed ring node 3 does not. Ring nodes 2 and 4 will each ultimately send a Neighbor_Status frame to the ring supervisor.

9-5.5.6 DLR Object

Devices that implement the DLR protocol shall implement the DLR Object, specified in Chapter 5.

9-5.6 Implementation Requirements

9-5.6.1 Embedded Switch Requirements and Recommendations

The following are general requirements and recommendations for all devices that implement embedded switch technology (whether implemented via commercially-available chips, FPGA, ASIC, etc.):

- IEEE 802.3 operation:
- Auto-negotiation, with 10/100Mbps, full/half duplex (Required)
- Forced setting of speed/duplex (Required)
- Auto MDIX (medium dependent interface crossover), in both auto-negotiate and forced speed/duplex modes. Note: This is a PHY and transformer issue, not an embedded switch issue. (Required)

- QoS:
  - 2 queues (Required); 4 (Recommended)
  - High priority queue for DLR frames, with strict priority scheduling for the high priority queue (Required)
  - Prioritization via 802.1Q/D (Required) and DSCP (highly recommended). Usage shall be consistent with the EtherNet/IP QoS scheme published in Volume 2. For IP frames the embedded switch should use the DSCP value. For non-IP frames the priority in the 802.1Q header should be used.
- Broadcast rate limiting for host CPU (Recommended). The broadcast threshold tolerated by a device is dependent on the host CPU. As a general recommendation, the broadcast rate limiting should be triggered when the broadcast traffic exceeds 1% of bandwidth.
- Filtering of incoming unicast and multicast to host CPU (Recommended, but in practice most all devices will require this).

9-5.6.2 DLR Implementation Requirements

The following implementation requirements apply to DLR nodes, whether ring supervisors or ring nodes:

- Preserve IEEE 802.1Q VLAN Id and tag priority of ring protocol frames
- Disable IP multicast filtering on ring ports or flush multicast filtering table of ring ports on ring state transitions.
- Configure multicast address for Beacon frames to be forwarded on ring ports, and to the host CPU for Beacon-based implementations.
- Configure multicast address for Announce and Locate_Fault frames to be forwarded to the host CPU and on ring ports.
- Configure multicast address for Neighbor_Check_Request /Response and Sign_On to be forwarded only to host CPU.
- Mechanism to flag the port through which such a frame was received from ring.
- Mechanism to forward such frames from host CPU on to ring only through the port it was intended to go out.
- Configure unicast MAC address of active ring supervisor so that supervisor frames forwarded on both ports
- Flush unicast MAC address tables on ring state transitions (or disable learning)
- Configure unicast MAC address of self so that it is not purged when MAC address table is flushed.
- Implement the Interface Counters and Media Counters attributes of the Ethernet Link Object (see Chapter 5) to aid in network monitoring.
- Implement the QoS Object with, at a minimum, DSCP marking of EtherNet/IP traffic generated by the device (see Chapter 5).
- Recommended: configure access control list or another suitable mechanism to remove device's own frames from network when received (e.g., during ring startup/restoration)
- Disable 802.3 and other hardware flow control mechanisms on ring ports

- The 802.3 methods to determine a link's presence when a port is configured to operate in forced speed/duplex mode frequently yield transient link state transitions when a cable is inserted. To prevent unnecessary ring state transitions during cable insertion, devices shall ensure that the link is stable before enabling frame forwarding to/from the associated port or generating a Link_Status frame. This may typically be accomplished by adding some debounce (e.g. a delay of 10-100 ms, depending on hardware implementation) to achieve stable link transitions.

9-5.6.3 IEEE 1588 / CIP Sync Considerations

In order to support applications requiring time synchronization (e.g., CIP Motion or CIP Sync), multi-port devices are recommended to support the following capabilities with respect to IEEE 1588/CIP Sync:

- Implement IEEE 1588 end-to-end transparent clock.
- Devices that also implement 1588 ordinary/boundary clock functionality should perform path delay measurement using Delay_Req/Delay_Resp frames whenever the ring state or network topology mode changes.
- Devices that implement 1588 ordinary/boundary clock functionality and are connected to the ring network indirectly should perform path delay measurement using Delay_Req/Delay_Resp frames per the ODVA-specific IEEE 1588 profile signaling message (refer to the Time Sync Object in Volume 1).

Devices not implementing these features will suffer from poor synchronization accuracy for short periods after a network reconfiguration.

Refer to the CIP Sync specification in Volume 1 for more information on CIP Sync.

9-5.6.4 IEEE 802.1D/802.1Q STP/RSTP/MSTP Considerations

In order for DLR to coexist with IEEE spanning tree protocols STP, RSTP and MSTP the active ring supervisor shall not forward multicast frames with destination address 01:80:C2:00:00:00 (BPDU frames) from one ring port to other, irrespective of ring state. However, if the active ring supervisor has non-ring ports (e.g., in the case of a switch) it shall forward said multicast frames between those non-ring ports and only between one ring port and those non-ring ports. This behavior shall be implemented to ensure that any loop through the ring other than the one through active ring supervisor is detected correctly by STP/RSTP/MSTP.

9-5.7 Using Non-DLR Nodes in the Ring Network

9-5.7.1 General Considerations

The DLR protocol does not, by design, require that all nodes in the ring implement the protocol. Non-DLR nodes may be placed in the ring, provided they support certain required capabilities as outlined in subsequent subsections.

The end user should be made aware that using non-DLR nodes in the ring can affect performance, lengthening the ring recovery times (see section 9-5.10 (Performance Analysis). For best performance, it is highly recommended that all nodes in the ring implement the DLR protocol.

Where the use of non-DLR devices cannot be avoided, it is highly recommended that such devices be connected to the network via a DLR-aware device such as a 3-port switch (2 ports connected to the network, 1 port connected to the non-DLR device).

When using non-DLR nodes directly in the ring, certain capabilities and/or configuration steps are required of those nodes in order for the DLR protocol to function properly. These capabilities and configuration steps are described further in the following sections.

9-5.7.2 Non-DLR End Devices

Examples of non-DLR devices might include an existing 2-port I/O module or drive that supports embedded switch technology but has not implemented the DLR protocol. Such devices may be inserted directly into the ring network. However in order to ensure proper functioning of the ring, the non-DLR end device must have the following capabilities:

- Disable unicast MAC address learning (or not employ MAC address learning at all). Since Beacon frames will arrive on both ports with the supervisor's MAC address, address learning will cause the supervisor's MAC to bounce from one port to the next. If the supervisor is also an I/O device, the I/O communications can be interrupted.
- Disable multicast filtering on DLR ring ports. If not disabled, multicast messages may not be forwarded to other ring nodes.
- Support reception of 802.1Q frames and preserve VLAN Id and tag priority. DLR messages may be dropped if not supported, or not queued properly if tag priority is not preserved.
- Implement priority queues with highest priority queue used for DLR messages, with strict priority scheduling. If not supported, ring recovery performance may be affected.

It is the end user's responsibility to ensure that the non-DLR device supports the above capabilities.

9-5.7.3 Non-DLR Switches

It is possible for commercially-available managed Ethernet switches that are not DLR aware to be placed directly in the ring. For example, a user may wish to connect a ring of end devices into a switch in order to connect the end devices into the user's larger network.

In order to simplify the switch configuration, it is advisable to connect the non-DLR switch to the ring via a DLR-aware device such as a simple 3-port switch. When the non-DLR switch is connected directly into the ring, a number of configuration steps are required. The specifics of the configuration steps may vary from vendor to vendor. In general, the configuration steps required are outlined in the subsequent two sections.

Note: Using non-DLR switches can result in loss of unicast frames for some period of time following a ring fault or restoration. After a ring fault/restoration, the switch's MAC learning tables may be invalid as devices are now reachable through different ports. Until the MAC learning tables are updated as a result of devices sending frames, unicast frames may not reach the target devices.

For an EtherNet/IP I/O connection, typically one cyclic production cycle will be lost. In some circumstances, more than a single production may be lost. For example, in the case where the T-O RPI is slower than the O-T RPI, O-T packets will not be delivered until the target device sends a packet to update the switch's MAC table. Or, if the device is a polled device (such as an explicit message server), or has only unidirectional connections, the device may not generate packets to update the switch's MAC learning table.

In such cases where undesirable unicast packet loss occurs, the user has several options:

1. Configure static unicast MAC addresses for the switch ports connected to the ring

2. Disable unicast MAC learning in the switch

3. Use a DLR-capable device such as a 3-port switch to connect the larger switch to the ring

As described in section 9-5.2, DLR protocol rings can be used in topologies with switches that are running IEEE Spanning Tree Protocols (STP, RSTP, MSTP). In the most common configurations, as shown in section 9-5.1, DLR rings can be connected to such managed switches without issue.

The DLR protocol does not support any non-DLR loop(s) passing through portions of a DLR ring, irrespective of whether a switch supports DLR or not and irrespective of whether it supports Spanning Tree Protocols. Figure 9-5.7 and Figure 9-5.8 show examples of unsupported DLR topologies, with non-DLR loops traversing the DLR ring. Such topologies can cause erratic network behavior and can be resolved manually by removing one or more of non-DLR redundant links.

Figure 9-5.7 Unsupported Topology, Example 1

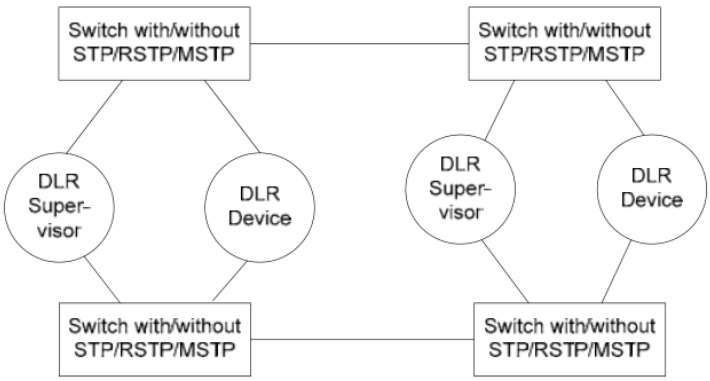
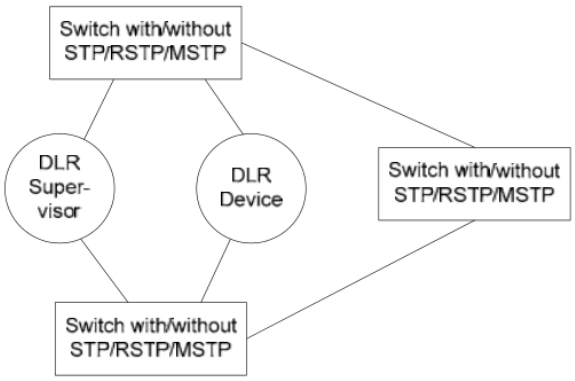


Figure 9-5.8 Unsupported Topology, Example 2



9-5.7.3.1 Switch Configuration For Non-VLAN Rings

This section outlines the general procedure to configure a managed switch for a non-VLAN based ring network. Actual commands to configure a specific switch is switch dependant, but can be deduced from the procedure below.

1. Configure quality of service (QoS) to ensure deterministic behavior and high performance for ring operation. The switch should support at least two queues, preferably four, and must support strict priority scheduling for the highest priority queue. The highest priority queue on ports connected to ring network must be configured for ring protocol frames (802.1Q priority 7). Note that while it is acceptable to share highest priority queue with IEEE 1588 PTP event messages, it is strongly recommended that no other traffic be shared on this queue. For IP frames the switch should use the DSCP value. For non-IP frames the priority in the 802.1Q header should be used.

2. If required, configure the two ports of switch connected ring network to preserve IEEE 802.1Q tag priority of ring protocol frames when they pass through the ports. Most switches will need no specific configuration for this step.

3. Disable IP multicast filtering on the two ports of switch connected to ring. This step must be done to facilitate uninterrupted delivery of EtherNet/IP multicast connection data after a ring reconfiguration. Some switches provide a direct way to configure forwarding of all IP multicast traffic on a per port basis. Other switches provide a way to configure designation of individual ports as multicast router ports. Both methods achieve the same effect of disabling multicast filtering on specific ports. See RFC4541 -- Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches for details.

4. Statically configure the three multicast addresses used by ring protocol to be forwarded only on two ports of switch connected to ring. This step must be done to prevent multicast ring protocol frames from being forwarded on other ports of switch.

5. Configure unicast MAC addresses of all configured ring supervisors statically into the MAC table of switch such that unicast traffic destined for ring supervisors will be forwarded through both ports of switch connected to ring. This step must be done to prevent switch from getting confused by bi-directional ring beacons from active ring supervisor.

9-5.7.3.2 Switch Configuration for a VLAN-based Ring

This section outlines the general procedure to configure a managed switch for a VLAN-based ring network. The actual configuration commands are switch-dependant, but can be deduced from the procedure below. Assume all ring supervisors have been configured to use VLAN ID *ring_vlan_id* for ring protocol frames. Note that the *ring_vlan_id* will be used only for ring protocol frames. Assume that the switch will use VLAN ID *default_vlan_id* for all untagged frame traffic including EtherNet/IP frames.

1. Configure quality of service (QoS) to ensure deterministic behavior and high performance for ring operation. The switch should support at least two queues, preferably four, and must support strict priority scheduling for the highest priority queue. The highest priority queue on ports connected to ring network must be configured for ring protocol frames (802.1Q priority 7). Note that while it is acceptable to share highest priority queue with IEEE 1588 PTP event messages, it is strongly recommended that no other traffic be shared on this queue. For IP frames the switch should use the DSCP value. For non-IP frames the priority in the 802.1Q header should be used.

2. Create VLAN's for *ring_vlan_id* and *default_vlan_id* on switch.

3. Configure the two ports of switch connected ring to participate in VLAN's *ring_vlan_id* and *default_vlan_id.* The two ports should be configured such that traffic on *default_vlan_id* should go untagged on egress and incoming untagged traffic should be assigned to *default_vlan_id*. The two ports should be configured to preserve VLAN tag on *ring_vlan_id* when ring protocol frames pass through the ports. Care must be taken to ensure that none of the other ports on switch participate in VLAN *ring_vlan_id*.

4. Disable IP multicast filtering on the two ports of switch connected to ring, but only for VLAN *default_vlan_id.* This step must be done to facilitate bump less delivery of EtherNet/IP multicast connection data after a ring reconfiguration. Some switches provide a direct way to configure forwarding of all IP multicast traffic on a per port basis. Other switches provide a way to configure designation of individual ports as multicast router ports. Both methods achieve the same effect of disabling multicast filtering on specific ports. See RFC4541 -- Considerations for Internet Group Management Protocol (IGMP) and Multicast Listener Discovery (MLD) Snooping Switches for details.

5. Configure unicast MAC addresses of all configured ring supervisors statically into the MAC table of switch such that unicast traffic destined for ring supervisors will be forwarded through both ports of switch connected to ring. This configuration must be done for both VLAN's *ring_vlan_id* and *default_vlan_id*. This step must be done to prevent switch from getting confused by bi-directional ring beacons from active ring supervisor. This step may be omitted, but will result in minor loss of performance.

9-5.8 DLR Messages

9-5.8.1 General

The following sections specify the DLR protocol messages. Several items of importance should be noted:

1. DLR messages are sent using the IEEE 802.1Q frame format. Messages shall be transmitted using the highest priority (7), which shall be preserved as the DLR frames are transferred through the LAN.

2. To distinguish the DLR frame data types from CIP data types, the UINTx convention is used to indicate an x-bit, unsigned integer value in the tables below. Multi-byte data types within the DLR frames shall be encoded using big-endian byte ordering.

3. The following destination MAC addresses are used for DLR messages:

Table 9-5.2 DLR MAC Address Usage

MAC Address	Usage
01-21-6C-00-00-01	Beacon
01-21-6C-00-00-02	Neighbor_Check_Request, Neighbor_Check_Response, Sign_On
01-21-6C-00-00-03	Announce, Locate_Fault
MAC address of active ring supervisor	Link_Status / Neighbor_Status

9-5.8.2 Common Frame Header

DLR messages shall be sent using the IEEE 802.1Q frame format, as shown below:

Table 9-5.3 Common Frame Header Format

Byte	Field	Type	Remarks
0	Destination MAC Address	UINT8[6]	
6	Source MAC Address	UINT8[6]	
12	802.1Q Tag Type	UINT16	= 0x8100
14	802.1Q Tag control	UINT16	= 0xE000 + VLAN_ID (default value=0)
16	Ring EtherType	UINT16	= 0x80E1
18	Ring Sub-type	UINT8	= 0x02
19	Ring Protocol Version	UINT8	= 0x01

The following common fields are used in the DLR message payload:

Table 9-5.4 DLR Message Payload Fields

Byte	Field	Type	Remarks
20	Frame Type	UINT8	Identifies the DLR message type: 0x01 – Beacon 0x02 – Neighbor_Check_Request 0x03 – Neighbor_Check_Response 0x04 – Link_Status / Neighbor_Status 0x05 – Locate_Fault 0x06 – Announce 0x07 – Sign_On
21	Source Port	UINT8	Identifies the port on the device through which this message originated: 0x00 – Port 1 or Port 2 0x01 – Port 1 0x02 – Port 2
22	Source IP Address	UINT32	Source IP address of the sending node. If none is configured, the value shall be 0.
26	Sequence Id	UINT32	DLR message sequence Id. When originating a message a node shall increment the Sequence Id by 1. When responding to a message (Neighbor_Check response and Sign_On), a node shall use the Sequence Id from the original request message.
30	Dependent on DLR message type		See Tables below
…	FCS	UINT32	IEEE 802.3 FCS

9-5.8.3 Beacon Frame

The Beacon frame is sent by the active ring supervisor to determine the health of the ring topology.

Table 9-5.5 shows the format of the Beacon frame:

Table 9-5.5 Beacon Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x01
21	Source Port	UINT8	= 0x0
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	
30	Ring State	UINT8	See Table 9-5.6 below
31	Supervisor Precedence	UINT8	From the DLR Object
32	Beacon Interval	UINT32	From the DLR Object
36	Beacon Timeout	UINT32	In Microseconds
40	Reserved	UINT8[20]	Sender shall set to zero, receiver shall ignore
60	FCS	UINT32	

Table 9-5.6 Ring State Values

Ring State	Value
RING_NORMAL_STATE	0x01
RING_FAULT_STATE	0x02

9-5.8.4 Neighbor_Check_Request

The Neighbor_Check_Request is sent by a ring node as a result of receiving a Locate_Fault frame, and by an active ring supervisor when Beacon loss has been detected.

Table 9-5.7 Neighbor_Check_Request Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x02
21	Source Port	UINT8	= 0x1 or 0x2
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	
30	Reserved	UINT8[30]	Sender shall set to zero, receiver shall ignore
60	FCS	UINT32	

9-5.8.5 Neighbor_Check_Response

The Neighbor_Check_Response frame is sent in response to the Neighbor_Check_Request.

Table 9-5.8 Neighbor_Check_Response Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x03
21	Source Port	UINT8	= 0x1 or 0x2
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	= Sequence Id of Neighbor_Check_Request
30	Request Source Port	UINT8	= Source Port of Neighbor_Check_Request
31	Reserved	UINT8[29]	Sender shall set to zero, receiver shall ignore
60	FCS	UINT32	

9-5.8.6 Link_Status/Neighbor_Status

A Link_Status frame is sent when a ring node has detected a link failure or in response to a Locate_Fault frame when one of its ports is not active. A Neighbor_Status frame is sent when the Neighbor Check process has timed out. These frames share the same format, differentiated by the Link/Neighbor Status Flag.

Table 9-5.9 Link_Status/Neighbor_Status Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x04
21	Source Port	UINT8	= 0
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	
30	Link/Neighbor Status	UINT8	Interpreted as a bit map, see Table 9-5.10 below
31	Reserved	UINT8[29]	Sender shall set to zero, receiver shall ignore
60	FCS	UINT32	

Table 9-5.10 Link/Neighbor Status Values

Bit(s):	Called	Definition
0 (least significant)	Port 1 Active	Set to 1 if node's port 1 is active
1	Port 2 Active	Set to 1 if node's port 2 is active
2-6	Reserved	Sender shall set to zero, receiver shall ignore
7	Link/Neighbor Status Flag	Set to 0 if frame is Link_Status frame Set to 1 if frame is Neighbor_Status frame

9-5.8.7 Locate_Fault

The Locate_Fault frame is sent by the active ring supervisor as a result of loss of Beacon frames, and as a result of the Verify_Fault_Location service sent to the DLR Object.

Table 9-5.11 Locate_Fault Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x05
21	Source Port	UINT8	= 0x0
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	
30	Reserved	UINT8[30]	Sender shall set to zero, receiver shall ignore
60	FCS	UINT32	

9-5.8.8 Announce

The Announce frame is sent by the active ring supervisor to indicate a change in the ring state, and to notify the ring topology to announce-based nodes.

Table 9-5.12 Announce Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x06
21	Source Port	UINT8	= 0x0
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	
30	Ring State	UINT8	As in the Beacon frame
31	Reserved	UINT8[29]	Sender shall set to zero, receiver shall ignore
60	FCS	UINT32	

9-5.8.9 Sign_On

The Sign_On frame is sent initially by the active ring supervisor, in order to identify all participating ring nodes. The receiving ring participant node traps the Sign_On frame and forwards it only to the host CPU. The host CPU increments the number of nodes in list, add its own addresses to list and transmits the Sign_On frame only through the other port than the receiving port.

Table 9-5.13 Sign_On Frame Format

Byte	Field	Type	Remarks
20	Frame Type	UINT8	= 0x07
21	Source Port	UINT8	= 0x1 or 0x2
22	Source IP Address	UINT32	= 0x0, if source has no IP address
26	Sequence Id	UINT32	
30	Number of Nodes in List	UINT16	
32	Node 1 MAC Address	UINT8[6]	Node 1 is always the active supervisor
38	Node 1 IP Address	UINT32	= 0x0, if no IP address
42	Node 2 MAC Address	UINT8[6]	
48	Node 2 IP Address	UINT32	= 0x0, if no IP address
…	Reserved	UINT8[…]	Sender shall set to zero, receiver shall ignore
N	FCS	UINT32	N should be padded to at least 60, if needed

9-5.9 State Diagrams and Event Matrixes

9-5.9.1 Beacon-Based Ring Node

Figure 9-5.9 State Transition Diagram for Beacon Frame Based Non-Supervisor Ring Node

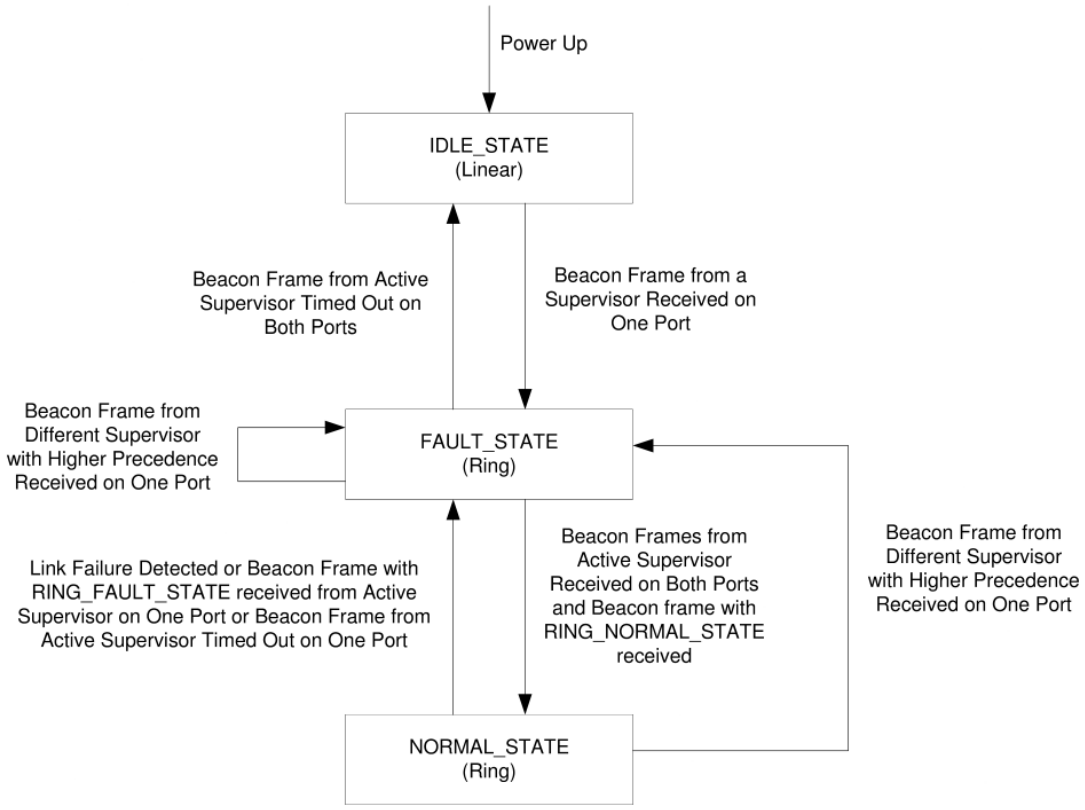


Table 9-5.14 Parameter Values for Beacon frame based Non-Supervisor Ring Node

Parameter	Value
Active supervisor precedence	Obtained from Beacon frame
Active supervisor MAC address	Obtained from Beacon frame
Ring protocol VLAN ID	Obtained from Beacon frame
Beacon timeout duration	Obtained from Beacon frame
Neighbor check timeout duration	100 milliseconds
Maximum retry limit for Neighbor_Check_Request frame	3 (total number of tries)

The following notes apply to the State-Event-Action table:

- LastBcnRcvPort is a bit string variable used to track port(s) on which last Beacon frame was received, with the following bit definitions:

Bit	Description
0	Set if a beacon frame from the active supervisor has been received on Port 1
1	Set if a beacon frame from the active supervisor has been received on Port 2
2-N	Not used; set to 0

- MAC address 11-22-33-44-55-66 shall be encoded as 0x112233445566 for numerical comparison in case of supervisor precedence tie.
- Nodes shall statically configure unicast MAC address of current active ring supervisor into unicast MAC learning table to be forwarded only through both ring ports.

Table 9-5.15 State-Event-Action Table for Beacon Frame Based Non-Supervisor Ring Node

Event No.	Current State	Event	Action(s)
1	None	Power up	Initialize and transition to IDLE_STATE
2	IDLE_STATE	Beacon frame from a ring supervisor received on port 1	Save as active supervisor precedence, MAC address, VLAN ID and beacon timeout duration Set LastBcnRcvPort to 1 Start beacon timeout timer for port 1 Transition to FAULT_STATE and flush unicast MAC address learning table
3	IDLE_STATE	Beacon frame from a ring supervisor received on port 2	Save as active supervisor precedence, MAC address, VLAN ID and beacon timeout Set LastBcnRcvPort to 2 Start beacon timeout timer for port 2 Transition to FAULT_STATE and flush unicast MAC address learning table
4	IDLE_STATE	Link lost on port 1	Stop forwarding frames on port 1
5	IDLE_STATE	Link lost on port 2	Stop forwarding frames on port 2
6	IDLE_STATE	Link restored on port 1	Start forwarding frames on port 1
7	IDLE_STATE	Link restored on port 2	Start forwarding frames on port 2
8	IDLE_STATE	Frame from self received on port 1 or port 2	Do not forward frame on network and drop it (If capable of doing so)
9	IDLE_STATE	Locate_Fault or Neighbor_Check_Request or Neighbor_Check_Response or Sign_On frame received on port 1 or port 2	Ignore
10	FAULT_STATE	Beacon frame from active ring supervisor received on port 1 and LastBcnRcvPort is 1	Restart port 1 beacon timeout timer
11	FAULT_STATE	Beacon frame from active ring supervisor received on port 2 and LastBcnRcvPort is 2	Restart port 2 beacon timeout timer

Event No.	Current State	Event	Action(s)
12	FAULT_STATE	Beacon timeout timer expired for port 1 and LastBcnRcvPort is 1	Stop neighbor check timeout timers for both ports, transition to IDLE_STATE and flush unicast MAC address learning table
13	FAULT_STATE	Beacon timeout timer expired for port 2 and LastBcnRcvPort is 2	Stop neighbor check timeout timers for both ports, transition to IDLE_STATE and flush unicast MAC address learning table
14	FAULT_STATE	Beacon frame from active ring supervisor received on port 2 and LastBcnRcvPort is 1 or 3	Set LastBcnRcvPort to 3 Start/restart port 2 beacon timeout timer If the ring state of beacon frame is not RING_NORMAL_STATE, return Else, stop neighbor check timeout timers for both ports Transition to NORMAL_STATE and flush unicast MAC address learning table
15	FAULT_STATE	Beacon frame from active ring supervisor received on port 1 and LastBcnRcvPort is 2 or 3	Set LastBcnRcvPort to 3 Start/restart port 1 beacon timeout timer If the ring state of beacon frame is not RING_NORMAL_STATE, return Else, stop neighbor check timeout timers for both ports Transition to NORMAL_STATE and flush unicast MAC address learning table
16	FAULT_STATE	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 1	If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return Else, stop beacon timeout timers for both ports Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration Set LastBcnRcvPort to 1 Start beacon timeout timer for port 1 Stay in FAULT_STATE and flush unicast MAC address learning table
17	FAULT_STATE	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 2	If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return Else, stop beacon timeout timers for both ports Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration Set LastBcnRcvPort to 2 Start beacon timeout timer for port 2 Stay in FAULT_STATE and flush unicast MAC address learning table

Event No.	Current State	Event	Action(s)
18	FAULT_STATE	Link lost on port 1 and LastBcnRcvPort is 1	Stop port 1 beacon timeout timer and neighbor check timeout timers for both ports Transition to IDLE_STATE and flush unicast MAC address learning table
19	FAULT_STATE	Link lost on port 1 and LastBcnRcvPort is 2	Send Link_Status frame to active ring supervisor
20	FAULT_STATE	Link lost on port 2 and LastBcnRcvPort is 2	Stop port 2 beacon timeout timer and neighbor check timeout timers for both ports Transition to IDLE_STATE and flush unicast MAC address learning table
21	FAULT_STATE	Link lost on port 2 and LastBcnRcvPort is 1	Send Link_Status frame to active ring supervisor
22	FAULT_STATE	Link restored on port 1	Start forwarding frames on port 1
23	FAULT_STATE	Link restored on port 2	Start forwarding frames on port 2
24	FAULT_STATE	Locate_Fault frame received from active ring supervisor on port 1 or port 2	If link is not active on port 1 or port 2 send Link_Status frame to active ring supervisor Else clear neighbor status for port 1 and 2, send Neighbor_Check_Request for port 1 and 2, and start neighbor check timeout timer for port 1 and 2.
25	FAULT_STATE	Neighbor_Check_Request frame received on port 1	Send Neighbor_Check_Response frame on port 1
26	FAULT_STATE	Neighbor_Check_Request frame received on port 2	Send Neighbor_Check_Response frame on port 2
27	FAULT_STATE	Neighbor_Check_Response frame received on port 1	Stop neighbor check timeout timer for port 1 and save neighbor status for port 1
28	FAULT_STATE	Neighbor_Check_Response frame received on port 2	Stop neighbor check timeout timer for port 2 and save neighbor status for port 2
29	FAULT_STATE	Neighbor check timer timeout on port 1	If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 1 and start neighbor check timeout timer for port 1 Else, send Neighbor_Status frame to active ring supervisor
30	FAULT_STATE	Neighbor check timer timeout on port 2	If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 2 and start neighbor check timeout timer for port 2 Else, send Neighbor_Status frame to active ring supervisor
31	FAULT_STATE	Sign_On frame received on port 1 or port 2	Ignore
32	NORMAL_STATE	Link lost on port 1	Send Link_Status frame to active ring supervisor Stop beacon timeout timer for port 1 Set LastBcnRcvPort to 2, transition to FAULT_STATE and flush unicast MAC address learning table

Event No.	Current State	Event	Action(s)
33	NORMAL_STATE	Link lost on port 2	Send Link_Status frame to active ring supervisor Stop beacon timeout timer for port 2 Set LastBcnRcvPort to 2, transition to FAULT_STATE and flush unicast MAC address learning table
34	NORMAL_STATE	Beacon frame from active ring supervisor with RING_FAULT_STATE received on port 1	If the ring state of previous beacon frame received on port 1 was not RING_NORMAL_STATE, return Else, stop beacon timeout timer for port 2 Set LastBcnRcvPort to 1, transition to FAULT_STATE and flush unicast MAC address learning table
35	NORMAL_STATE	Beacon frame from active ring supervisor with RING_FAULT_STATE received on port 2	If the ring state of previous beacon frame received on port 2 was not RING_NORMAL_STATE, return Else, stop beacon timeout timer for port 1 Set LastBcnRcvPort to 2, transition to FAULT_STATE and flush unicast MAC address learning table
36	NORMAL_STATE	Beacon timeout timer expired for port 1	Set LastBcnRcvPort to 2, transition to FAULT_STATE and flush unicast MAC address learning table
37	NORMAL_STATE	Beacon timeout timer expired for port 2	Set LastBcnRcvPort to 1, transition to FAULT_STATE and flush unicast MAC address learning table
38	NORMAL_STATE	Beacon frame from active ring supervisor received on port 1	Restart port 1 beacon timeout timer
39	NORMAL_STATE	Beacon frame from active ring supervisor received on port 2	Restart port 2 beacon timeout timer
40	NORMAL_STATE	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 1	If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return Else, stop beacon timeout timers for both ports Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration Set LastBcnRcvPort to 1 Start beacon timeout timer for port 1 Transition to FAULT_STATE and flush unicast MAC address learning table

Event No.	Current State	Event	Action(s)
41	NORMAL_STATE	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 2	If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return Else, stop beacon timeout timers for both ports Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration Set LastBcnRcvPort to 2 Start beacon timeout timer for port 2 Transition to FAULT_STATE and flush unicast MAC address learning table
42	NORMAL_STATE	Locate_Fault frame received from active ring supervisor on port 1 or port 2	Ignore
43	NORMAL_STATE	Neighbor_Check_Request or Neighbor_Check_Response frame received on port 1 or port 2	Ignore
44	NORMAL_STATE	Sign_On frame received on port 1	Add node data to participant list and send frame through port 2
45	NORMAL_STATE	Sign_On frame received on port 2	Add node data to participant list and send frame through port 1

Notes:
Network Topology attribute shall be updated at events 1, 2, 3, 12, 13, 18 and 20 to appropriate value.

Network Status attribute shall be updated at events 1, 2, 3, 8, 12, 13, 14, 15, 18, 20, 32, 33, 34, 35, 36, 37, 40 and 41 to appropriate value.

9-5.9.2 Announce-Based Ring Node

Figure 9-5.10 State Transition Diagram for Announce Frame Based Non-Supervisor Ring Node

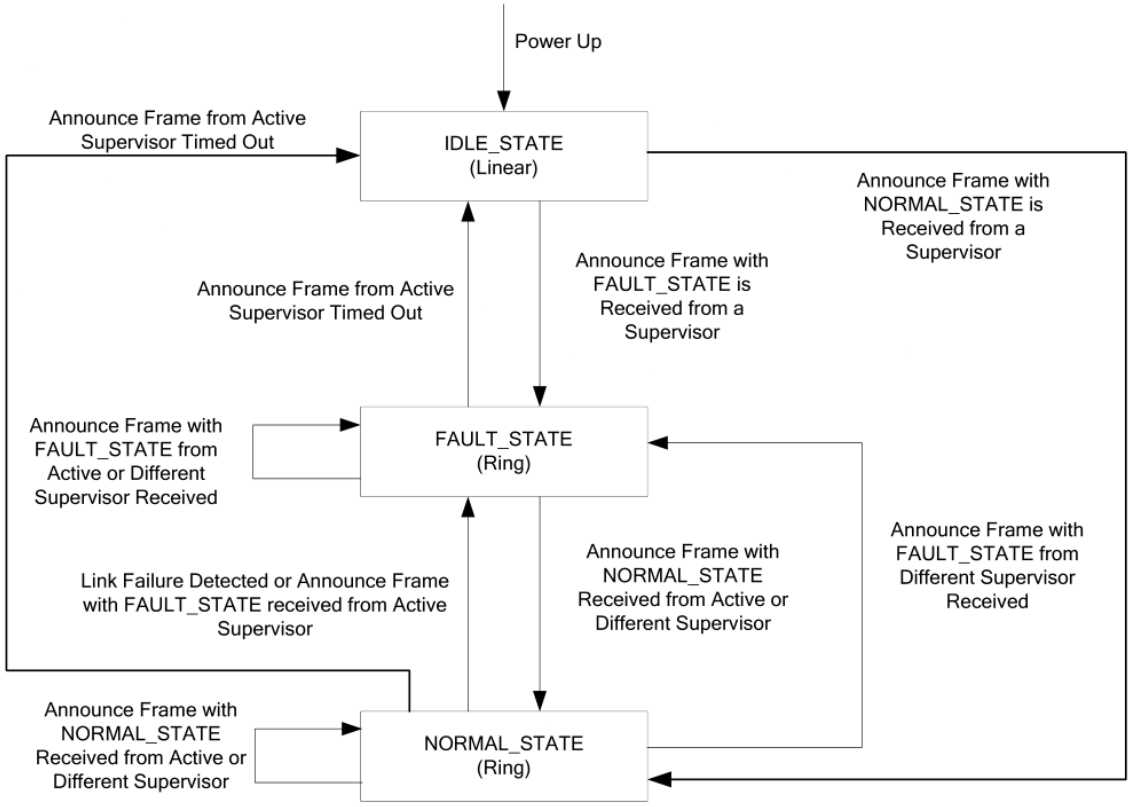


Table 9-5.16 Parameter Values for Announce frame based Non-Supervisor Ring Node

Parameter	Value
Active supervisor MAC address	Obtained from Announce frame
Ring protocol VLAN ID	Obtained from Announce frame
Announce timeout duration	2.5 seconds
Neighbor check timeout duration	100 milliseconds
Maximum retry limit for Neighbor_Check_Request frame	3 (total number of tries)

The following notes apply to the State-Event-Action table:

- Sequence count must be verified for all Announce frames from the active ring supervisor using modular unsigned 32-bit integer arithmetic as described in Chapter 3-4.2, CIP Transport Class 0 and Class 1 Packet Ordering to deal with sequence number rollover. Only Announce frames with sequence count greater than last Announce frame should be processed. Others must be silently discarded.
- Nodes shall statically configure unicast MAC address of current active ring supervisor into unicast MAC learning table to be forwarded only through both ring ports.
- Nodes shall statically configure multicast MAC address of beacon frames into MAC learning table to be forwarded only through both ring ports and not to CPU.

Table 9-5.17 State-Event-Action Table for Announce Frame Based Non-Supervisor Ring Node

Event No.	Current State	Event	Action(s)
1	None	Power up	a. Initialize and transition to IDLE_STATE
2	IDLE_STATE	Announce frame from a ring supervisor received on port 1 or port 2	a. Save as active supervisor MAC address, VLAN ID and sequence count b. Start announce timeout timer c. Transition to ring state in Announce frame and flush unicast MAC address learning table
3	IDLE_STATE	Link lost on port 1	a. Stop forwarding frames on port 1
4	IDLE_STATE	Link lost on port 2	a. Stop forwarding frames on port 2
5	IDLE_STATE	Link restored on port 1	a. Start forwarding frames on port 1
6	IDLE_STATE	Link restored on port 2	a. Start forwarding frames on port 2
7	IDLE_STATE	Frame from self received on port 1 or port 2	a. Do not forward frame on network and drop it (If capable of doing so)
8	IDLE_STATE	Locate_Fault or Neighbor_Check_Request or Neighbor_Check_Response or Sign_On frame received on port 1 or port 2	a. Ignore
9	FAULT_STATE	Announce frame from active ring supervisor received on port 1 or port 2 with higher sequence count than last Announce frame	a. Save sequence count b. Update VLAN ID (if different) c. Restart announce timeout timer d. If ring state in Announce frame is RING_NORMAL_STATE and links on both ports are active, stop neighbor check timeout timers for both ports, transition to NORMAL_STATE and flush unicast MAC address learning table e. Else, stay in FAULT_STATE
10	FAULT_STATE	Announce timeout timer expired	a. Stop neighbor check timeout timers for both ports, transition to IDLE_STATE and flush unicast MAC address learning table
11	FAULT_STATE	Announce frame from different ring supervisor MAC address than active ring supervisor is received on port 1 or port 2	a. Restart announce timeout timer and stop neighbor check timeout timers for both ports b. Save new active supervisor ,MAC address, VLAN ID and sequence count c. If ring state in Announce frame is RING_NORMAL_STATE and links on both ports are active, transition to NORMAL_STATE and flush unicast MAC address learning table d. Else, stay in FAULT_STATE

Event No.	Current State	Event	Action(s)
12	FAULT_STATE	Link lost on port 1	a. If link on port 2 is active, send Link_Status frame to active ring supervisor b. Else, stop announce timeout timer, stop neighbor check timeout timers for both ports, transition to IDLE_STATE and flush unicast MAC address learning table
13	FAULT_STATE	Link lost on port 2	a. If link on port 1 is active, send Link_Status frame to active ring supervisor b. Else, stop announce timeout timer, stop neighbor check timeout timers for both ports, transition to IDLE_STATE and flush unicast MAC address learning table
14	FAULT_STATE	Link restored on port 1	a. Start forwarding frames on port 1
15	FAULT_STATE	Link restored on port 2	a. Start forwarding frames on port 2
16	FAULT_STATE	Locate_Fault frame received from active ring supervisor	a. If links are active on both ports, clear neighbor status for both ports, send Neighbor_Check_Request frame on both ports and start neighbor check timeout timer for both ports b. Else send Link_Status frame to active ring supervisor
17	FAULT_STATE	Neighbor_Check_Request frame received on port 1	a. Send Neighbor_Check_Response frame on port 1
18	FAULT_STATE	Neighbor_Check_Request frame received on port 2	a. Send Neighbor_Check_Response frame on port 2
19	FAULT_STATE	Neighbor_Check_Response frame received on port 1	a. Stop neighbor check timeout timer for port 1 and save neighbor status for port 1
20	FAULT_STATE	Neighbor_Check_Response frame received on port 2	a. Stop neighbor check timeout timer for port 2 and save neighbor status for port 2
21	FAULT_STATE	Neighbor check timer timeout on port 1	a. If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 1 and start neighbor check timeout timer for port 1 b. Else, send Neighbor_Status frame to active ring supervisor
22	FAULT_STATE	Neighbor check timer timeout on port 2	a. If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 2 and start neighbor check timeout timer for port 2 b. Else, send Neighbor_Status frame to active ring supervisor
23	FAULT_STATE	Sign_On frame received on port 1 or port 2	a. Ignore

Event No.	Current State	Event	Action(s)
24	NORMAL_STATE	Link lost on port 1	a. Send Link_Status frame to active ring supervisor b. Transition to FAULT_STATE and flush unicast MAC address learning table
25	NORMAL_STATE	Link lost on port 2	a. Send Link_Status frame to active ring supervisor b. Transition to FAULT_STATE and flush unicast MAC address learning table
26	NORMAL_STATE	Announce frame from active ring supervisor received on port 1 or port 2 with higher sequence count than last Announce frame	a. Save sequence count b. Update VLAN ID (if different) c. Restart announce timeout timer d. If ring state in Announce frame is RING_FAULT_STATE, transition to FAULT_STATE and flush unicast MAC address learning table e. Else, stay in NORMAL_STATE
27	NORMAL_STATE	Announce timeout timer expired	a. Transition to IDLE_STATE and flush unicast MAC address learning table
28	NORMAL_STATE	Announce frame from different ring supervisor MAC address than active ring supervisor is received on port 1 or port 2	a. Restart announce timeout timer b. Save new active supervisor MAC address, VLAN ID and sequence count c. If ring state in Announce frame is RING_FAULT_STATE, transition to FAULT_STATE and flush unicast MAC address learning table d. Else, stay in NORMAL_STATE
29	NORMAL_STATE	Locate_Fault frame received from active ring supervisor	a. Ignore
30	NORMAL_STATE	Neighbor_Check_Request or Neighbor_Check_Response frame received on port 1 or port 2	a. Ignore
31	NORMAL_STATE	Sign_On frame received on port 1	a. Add node data to participant list and send frame through port 2
32	NORMAL_STATE	Sign_On frame received on port 2	a. Add node data to participant list and send frame through port 1

Notes:

- Network Topology attribute shall be updated at events 1, 2, 10, 12, 13 and 27 to appropriate value.
- Network Status attribute shall be updated at events 1, 2, 7, 9, 10, 11, 12, 13, 24, 25, 26, 27 and 28 to appropriate value.

9-5.9.3 Ring Supervisor

Figure 9-5.11 State Transition Diagram for Ring Supervisor

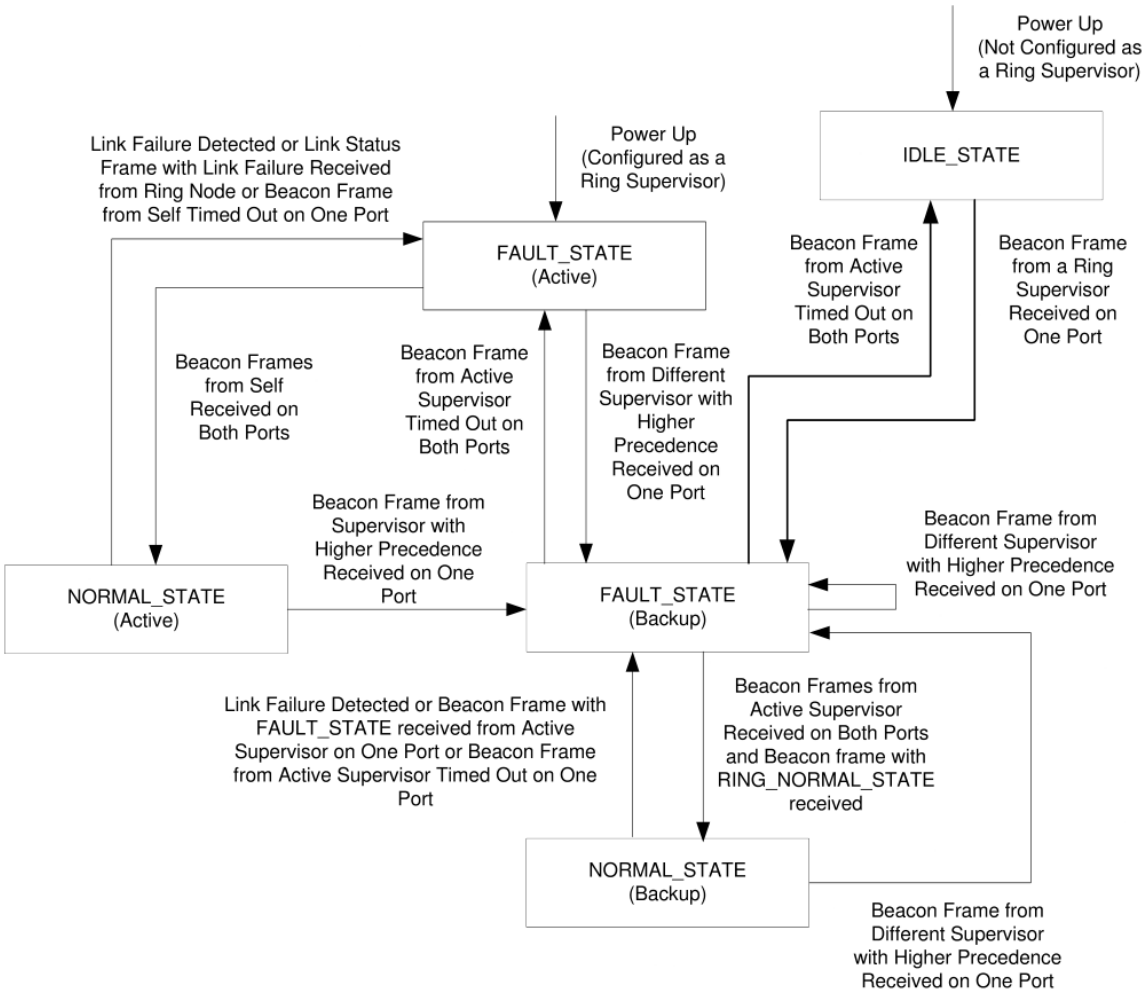


Table 9-5.18 Parameter Values for Ring Supervisor Node

Parameter	Value
Active supervisor precedence	Default 0 or as configured
Active supervisor MAC address	As manufactured
Ring protocol VLAN ID	Default 0 or as configured
Beacon interval duration	Default 400 microseconds or as configured
Beacon timeout duration	Default 1960 microseconds or as configured
Announce suppression duration	2 times Beacon Timeout
Announce interval duration	1 second
Active supervisor precedence (Backup)	Obtained from Beacon frame
Active supervisor MAC address (Backup)	Obtained from Beacon frame
Ring protocol VLAN ID (Backup)	Obtained from Beacon frame
Beacon timeout duration (Backup)	Obtained from Beacon frame
Neighbor check timeout duration	100 milliseconds
Maximum retry limit for Neighbor_Check_Request frame	3 (total number of tries)
Switchover quiet period duration during ring supervisor switchover	1 beacon timeout duration of last active ring supervisor
Sign on timeout duration	60 seconds

The following notes apply to the State-Event-Action table:

- LastBcnRcvPort is a bit string variable used to track port(s) on which last Beacon frame was received, with the following bit definitions:

Bit	Description
0	Set if a beacon frame from the active supervisor has been received on Port 1
1	Set if a beacon frame from the active supervisor has been received on Port 2
2-N	Not used; set to 0

- MAC address 11-22-33-44-55-66 shall be encoded as 0x112233445566 for numerical comparison in case of supervisor precedence tie.
- Nodes not configured as ring supervisor or acting as back up ring supervisor shall statically configure unicast MAC address of current active ring supervisor into unicast MAC learning table to be forwarded only through both ring ports.
- If the attributes of an active supervisor such as beacon interval duration, beacon timeout duration, precedence and VLAN ID are changed, then the active supervisor must follow special behavior before applying new values. If the active supervisor was in NORMAL_STATE (Active), then it should stop all timers, leave port 2 in blocked state and stop sending beacon and announce frames for 2 times (current) beacon timeout duration. If the active supervisor was in FAULT_STATE (Active), then it should stop all timers, leave port 2 in forwarding state and stop sending beacon and announce frames for 2 times (current) beacon timeout duration. At the end of the wait period it should start from event 1 in table below. Refer also to section 9-5.5.2.7 (Changing Ring Parameters).
- If a backup supervisor cannot support the currently active Beacon Interval and/or Beacon Timeout, the backup supervisor shall indicate the condition via the DLR Object, and shall not take over as active supervisor in the event of a ring reconfiguration.

- If the precedence attribute of a backup supervisor is reconfigured online to a higher value than current active ring supervisor, then it should stop all timers and start from event 1 in table below immediately. Refer also to section 9-5.5.2.7 (Changing Ring Parameters).
- If a non-supervisory is node is enabled as a ring supervisor online and the configured precedence value is higher than current active ring supervisor, then it should stop all timers and start from event 1 in table below immediately. For all other changes, it should just stay as a backup supervisor in current state.
- The active ring supervisor shall not forward multicast frames with destination address 01:80:C2:00:00:00 (BPDU) from one ring port to other, irrespective of ring state. However, if the active ring supervisor has non-ring ports, it shall forward said multicast frames between those non-ring ports and between only one ring port and those non-ring ports.

Table 9-5.19 State-Event-Action Table for Ring Supervisor Node

Event No.	Current State	Event	Action(s)
1	None	Power up	a. Initialize b. If not configured as ring supervisor transition to IDLE_STATE and return c. Else, set LastBcnRcvPort to 0 d. Enable forwarding of all frames on both ports e. Transition to FAULT_STATE (Active) f. Send Beacon frame through both ports and start beacon interval timer g. Start announce interval timer with announce suppression duration
2	IDLE_STATE	Beacon frame from a ring supervisor received on port 1	a. Save as active supervisor precedence, MAC address, VLAN ID and beacon timeout duration b. Set LastBcnRcvPort to 1 c. Start beacon timeout timer for port 1 d. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
3	IDLE_STATE	Beacon frame from a ring supervisor received on port 2	a. Save as active supervisor precedence, MAC address, VLAN ID and beacon timeout duration b. Set LastBcnRcvPort to 2 c. Start beacon timeout timer for port 2 d. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
4	IDLE_STATE	Link lost on port 1	a. Stop forwarding frames on port 1
5	IDLE_STATE	Link lost on port 2	a. Stop forwarding frames on port 2
6	IDLE_STATE	Link restored on port 1	a. Start forwarding frames on port 1
7	IDLE_STATE	Link restored on port 2	a. Start forwarding frames on port 2
8	IDLE_STATE	Frame from self received on port 1 or port 2	a. Do not forward frame on network and drop it

Event No.	Current State	Event	Action(s)
9	IDLE_STATE	Locate_Fault or Neighbor_Check_Request or Neighbor_Check_Response or Sign_On frame or Link_Status frame or Neighbor_Status frame received on port 1 or port 2 or Verify_Fault_Location service request received	a. Ignore
10	FAULT_STATE (Active)	Beacon interval timer expired	a. Send Beacon frame through both ports and start beacon interval timer
11	FAULT_STATE (Active)	Beacon frame from self received on port 1	a. Set LastBcnRcvPort to (LastBcnRcvPort | 0x1) b. Restart beacon timeout timer for port 1 c. If LastBcnRcvPort is not equal to 3, return d. Else, stop neighbor check timeout timers for both ports e. Block all traffic on port 2 except for special ring protocol frames f. Transition to NORMAL_STATE (Active) and flush unicast MAC address learning table g. Send Announce frame on port 1 and restart announce interval timer with announce interval duration h. Send Beacon frame on both ports and restart beacon interval timer i. Send Sign_On frame on port 1 and start sign on timeout timer
12	FAULT_STATE (Active)	Beacon frame from self received on port 2	a. Set LastBcnRcvPort to (LastBcnRcvPort | 0x2) b. Restart beacon timeout timer for port 2 c. If LastBcnRcvPort is not equal to 3, return d. Else, stop neighbor check timeout timers for both ports e. Block all traffic on port 2 except for special ring protocol frames f. Transition to NORMAL_STATE (Active) and flush unicast MAC address learning table g. Send Announce frame on port 1 and restart announce interval timer with announce interval duration h. Send Beacon frame on both ports and restart beacon interval timer i. Send Sign_On frame on port 1 and start sign on timeout timer

Event No.	Current State	Event	Action(s)
13	FAULT_STATE (Active)	Beacon frame from different supervisor MAC address than self received on port 1	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than self, drop new Beacon frame and return b. Else, stop beacon interval timer, stop beacon timeout timers for both ports, stop neighbor check timeout timers for both ports and stop announce interval timer c. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout d. Set LastBcnRcvPort to 1 e. Start beacon timeout timer for port 1 f. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
14	FAULT_STATE (Active)	Beacon frame from different supervisor MAC address than self received on port 2	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than self, drop new Beacon frame and return b. Else, stop beacon interval timer, stop beacon timeout timers for both ports, stop neighbor check timeout timers for both ports and stop announce interval timer c. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout d. Set LastBcnRcvPort to 2 e. Start beacon timeout timer for port 2 f. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
15	FAULT_STATE (Active)	Beacon timeout timer expired for port 1	a. Set LastBcnRcvPort to (LastBcnRcvPort & ~0x1)
16	FAULT_STATE (Active)	Beacon timeout timer expired for port 2	a. Set LastBcnRcvPort to (LastBcnRcvPort & ~0x2)
17	FAULT_STATE (Active)	Link lost on port 1	a. Save self as last active node on port 1 b. Stop forwarding frames on port 1
18	FAULT_STATE (Active)	Link lost on port 2	a. Save self as last active node on port 2 b. Stop forwarding frames on port 2
19	FAULT_STATE (Active)	Link restored on port 1	a. Start forwarding frames on port 1
20	FAULT_STATE (Active)	Link restored on port 2	a. Start forwarding frames on port 2
21	FAULT_STATE (Active)	Link_Status frame received on port 1	a. If Link_Status frame does not report a link failure, drop frame and return b. Else, save source node as last active node on port 1

Event No.	Current State	Event	Action(s)
22	FAULT_STATE (Active)	Link_Status frame received on port 2	a. If Link_Status frame does not report a link failure, drop frame and return b. Else, save source node as last active node on port 2
23	FAULT_STATE (Active)	Neighbor_Status frame received on port 1	a. If Neighbor_Status frame does not report a neighbor node failure, drop frame and return b. Else, save source node as last active node on port 1
24	FAULT_STATE (Active)	Neighbor_Status frame received on port 2	a. If Neighbor_Status frame does not report a neighbor node failure, drop frame and return b. Else save source node as last active node on port 2
25	FAULT_STATE (Active)	Announce interval timer expired	a. Send Announce frame through both ports and start announce interval timer with announce interval duration
26	FAULT_STATE (Active)	Verify_Fault_Location service request received	a. Send Locate_Fault frame on port 1 and port 2 b. If link is active on port 1, clear neighbor status for port 1, send Neighbor_Check_Request frame on port 1 and start neighbor check timeout timer for port 1 c. Else, save self as last active node on port 1 d. If link is active on port 2, clear neighbor status for port 2, send Neighbor_Check_Request frame on port 2 and start neighbor check timeout timer for port 2 e. Else, save self as last active node on port 2
27	FAULT_STATE (Active)	Neighbor_Check_Request frame received on port 1	a. Send Neighbor_Check_Response frame on port 1
28	FAULT_STATE (Active)	Neighbor_Check_Request frame received on port 2	a. Send Neighbor_Check_Response frame on port 2
29	FAULT_STATE (Active)	Neighbor_Check_Response frame received on port 1	a. Stop neighbor check timeout timer for port 1 and save neighbor status for port 1
30	FAULT_STATE (Active)	Neighbor_Check_Response frame received on port 2	a. Stop neighbor check timeout timer for port 2 and save neighbor status for port 2
31	FAULT_STATE (Active)	Neighbor check timer timeout on port 1	a. If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 1 and start neighbor check timeout timer for port 1 b. Else, save self as last active node on port 1

Event No.	Current State	Event	Action(s)
32	FAULT_STATE (Active)	Neighbor check timer timeout on port 2	a. If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 2 and start neighbor check timeout timer for port 2 b. Else, save self as last active node on port 2
33	FAULT_STATE (Active)	Sign_On frame received on port 1 or port 2	a. Ignore
34	NORMAL_STATE (Active)	Beacon interval timer expired	a. Send Beacon frame through both ports and start beacon interval timer
35	NORMAL_STATE (Active)	Beacon frame from self received on port 1	a. Restart beacon timeout timer for port 1
36	NORMAL_STATE (Active)	Beacon frame from self received on port 2	a. Restart beacon timeout timer for port 2
37	NORMAL_STATE (Active)	Beacon frame from different supervisor MAC address than self received on port 1	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return b. Else, stop beacon interval timer, stop beacon timeout timers for both ports, stop announce interval timer and stop sign on timeout timer c. Enable forwarding of all frames on port 2 d. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration e. Set LastBcnRcvPort to 1 f. Start beacon timeout timer for port 1 g. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table

Event No.	Current State	Event	Action(s)
38	NORMAL_STATE (Active)	Beacon frame from different supervisor MAC address than self received on port 2	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return b. Else, stop beacon interval timer, stop beacon timeout timers for both ports, stop announce interval timer and stop sign on timeout timer c. Enable forwarding of all frames on port 2 d. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration e. Set LastBcnRcvPort to 2 f. Start beacon timeout timer for port 2 g. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
39	NORMAL_STATE (Active)	Beacon timeout timer expired for port 1	a. Stop sign on timeout timer b. Set LastBcnRcvPort to 2 c. Enable forwarding of all frames on port 2 d. Transition to FAULT_STATE (Active) and flush unicast MAC address table e. Send Announce frame on both ports and restart announce interval timer with announce interval duration f. Send Beacon frame and restart beacon interval timer g. Send Locate_Fault frame h. Clear neighbor status for port 1 and 2, send Neighbor_Check_Request for port 1 and 2, and start neighbor check timeout timer for port 1 and 2
40	NORMAL_STATE (Active)	Beacon timeout timer expired for port 2	a. Stop sign on timeout timer b. Set LastBcnRcvPort to 1 c. Enable forwarding of all frames on port 2 d. Transition to FAULT_STATE (Active) and flush unicast MAC address table e. Send Announce frame on both ports and restart announce interval timer with announce interval duration f. Send Beacon frame and restart beacon interval timer g. Send Locate_Fault frame h. Clear neighbor status for port 1 and 2, send Neighbor_Check_Request for port 1 and 2, and start neighbor check timeout timer for port 1 and 2

Event No.	Current State	Event	Action(s)
41	NORMAL_STATE (Active)	Link lost on port 1	a. Save self as last active node on port 1 b. Stop sign on timeout timer c. Set LastBcnRcvPort to 2 d. Enable forwarding of all frames on port 2 e. Transition to FAULT_STATE (Active) and flush unicast MAC address table f. Send Announce frame on both ports and restart announce interval timer with announce interval duration g. Send Beacon frame and restart beacon interval timer
42	NORMAL_STATE (Active)	Link lost on port 2	a. Save self as last active node on port 2 b. Stop sign on timeout timer c. Set LastBcnRcvPort to 1 d. Enable forwarding of all frames on port 2 e. Transition to FAULT_STATE (Active) and flush unicast MAC address table f. Send Announce frame on both ports and restart announce interval timer with announce interval duration g. Send Beacon frame and restart beacon interval timer
43	NORMAL_STATE (Active)	Link_Status frame received on port 1	a. If Link_Status frame does not report a link failure, drop frame and return b. Else, save source node as last active node on port 1 c. Stop sign on timeout timer d. Enable forwarding of all frames on port 2 e. Transition to FAULT_STATE (Active) and flush unicast MAC address table f. Send Announce frame on both ports and restart announce interval timer with announce interval duration g. Send Beacon frame and restart beacon interval timer

Event No.	Current State	Event	Action(s)
44	NORMAL_STATE (Active)	Link_Status frame received on port 2	a. If Link_Status frame does not report a link failure, drop frame and return b. Else, save source node as last active node on port 2 c. Stop sign on timeout timer d. Enable forwarding of all frames on port 2 e. Transition to FAULT_STATE (Active) and flush unicast MAC address table f. Send Announce frame on both ports and restart announce interval timer with announce interval duration g. Send Beacon frame and restart beacon interval timer
45	NORMAL_STATE (Active)	Neighbor status frame received on port 1 or port 2	a. Ignore
46	NORMAL_STATE (Active)	Announce interval timer expired	a. Send Announce frame through port 1 and start announce interval timer with announce interval duration
47	NORMAL_STATE (Active)	Verify_Fault_Location service request received	a. Ignore
48	NORMAL_STATE (Active)	Neighbor_Check_Request or Neighbor_Check_Response frame received on port 1 or port 2	a. Ignore
49	NORMAL_STATE (Active)	Sign_On frame received on port 1 or port 2	a. If first entry in participant list is not self, drop frame and return b. Else, save ring participants count and participant list from frame and stop sign on timeout timer
50	NORMAL_STATE (Active)	Sign on timeout timer expired	a. Send Sign_On frame through port 1 and start sign on timeout timer
51	FAULT_STATE (Backup)	Beacon frame from active ring supervisor received on port 1 and LastBcnRcvPort is 1	a. Restart port 1 beacon timeout timer
52	FAULT_STATE (Backup)	Beacon frame from active ring supervisor received on port 2 and LastBcnRcvPort is 2	a. Restart port 2 beacon timeout timer

Event No.	Current State	Event	Action(s)
53	FAULT_STATE (Backup)	Beacon timeout timer expired for port 1 and LastBcnRcvPort is 1	a. If not configured as a ring supervisor, stop neighbor check timeout timers for both ports, transition to IDLE_STATE, flush unicast MAC address learning table and return b. Else, stop neighbor check timeout timers for both ports c. Wait for switchover quiet period duration d. Set LastBcnRcvPort to 0 e. Enable forwarding of all frames on both ports f. Transition to FAULT_STATE (Active) g. Send Beacon frame through both ports and start beacon interval timer h. Start announce interval timer with announce suppression duration
54	FAULT_STATE (Backup)	Beacon timeout timer expired for port 2 and LastBcnRcvPort is 2	a. If not configured as a ring supervisor, stop neighbor check timeout timers for both ports, transition to IDLE_STATE, flush unicast MAC address learning table and return b. Else, stop neighbor check timeout timers for both ports c. Wait for switchover quiet period duration d. Set LastBcnRcvPort to 0 e. Enable forwarding of all frames on both ports f. Transition to FAULT_STATE (Active) g. Send Beacon frame through both ports and start beacon interval timer h. Start announce interval timer with announce suppression duration
55	FAULT_STATE (Backup)	Beacon frame from active ring supervisor received on port 2 and LastBcnRcvPort is 1	a. Set LastBcnRcvPort to 3 b. Start/reset port 2 beacon timeout timer c. If the ring state of beacon frame is not RING_NORMAL_STATE, return d. Else, stop neighbor check timeout timers for both ports e. Transition to NORMAL_STATE (Backup) and flush unicast MAC address learning table

Event No.	Current State	Event	Action(s)
56	FAULT_STATE (Backup)	Beacon frame from active ring supervisor received on port 1 and LastBcnRcvPort is 2	a. Set LastBcnRcvPort to 3 b. Start/restart port 1 beacon timeout timer c. If the ring state of beacon frame is not RING_NORMAL_STATE d. Else, stop neighbor check timeout timers for both ports e. Transition to NORMAL_STATE (Backup) and flush unicast MAC address learning table
57	FAULT_STATE (Backup)	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 1	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return b. Else, stop beacon timeout timers for both ports c. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration d. Set LastBcnRcvPort to 1 e. Start beacon timeout timer for port 1 f. Stay in FAULT_STATE (Backup) and flush unicast MAC address learning table
58	FAULT_STATE (Backup)	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 2	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return b. Else, stop beacon timeout timers for both ports c. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration d. Set LastBcnRcvPort to 2 e. Start beacon timeout timer for port 2 f. Stay in FAULT_STATE (Backup) and flush unicast MAC address learning table
59	FAULT_STATE (Backup)	Link lost on port 1 and LastBcnRcvPort is 1	a. If not configured as a ring supervisor, stop neighbor check timeout timers for both ports, stop beacon timeout timers for both ports, transition to IDLE_STATE, flush unicast MAC address learning table and return b. Else, stop neighbor check timeout timers for both ports and stop beacon timeout timers for both ports c. Wait for switchover quiet period duration d. Set LastBcnRcvPort to 0 e. Enable forwarding of all frames on both ports

Event No.	Current State	Event	Action(s)
			f. Transition to FAULT_STATE (Active) g. Send Beacon frame through both ports and start beacon interval timer h. Start announce interval timer with announce suppression duration
60	FAULT_STATE (Backup)	Link lost on port 1 and LastBcnRcvPort is 2	a. Send Link Status frame to active ring supervisor
61	FAULT_STATE (Backup)	Link lost on port 2 and LastBcnRcvPort is 2	a. If not configured as a ring supervisor, stop neighbor check timeout timers for both ports, stop beacon timeout timers for both ports, transition to IDLE_STATE, flush unicast MAC address learning table and return b. Else, stop neighbor check timeout timers for both ports and stop beacon timeout timers for both ports c. Wait for switchover quiet period duration d. Set LastBcnRcvPort to 0 e. Enable forwarding of all frames on both ports f. Transition to FAULT_STATE (Active) g. Send Beacon frame through both ports and start beacon interval timer h. Start announce interval timer with announce suppression duration
62	FAULT_STATE (Backup)	Link lost on port 2 and LastBcnRcvPort is 1	a. Send Link Status frame to active ring supervisor
63	FAULT_STATE (Backup)	Link restored on port 1	a. Start forwarding frames on port 1
64	FAULT_STATE (Backup)	Link restored on port 2	a. Start forwarding frames on port 2
65	FAULT_STATE (Backup)	Locate_Fault frame received from active ring supervisor on port 1 or port 2	a. If link is not active on port 1 or port 2 send Link_Status frame to active ring supervisor b. Else clear neighbor status for port 1 and 2, send Neighbor_Check_Request for port 1 and 2, and start neighbor check timeout timer for port 1 and 2.
66	FAULT_STATE (Backup)	Neighbor_Check_Request frame received on port 1	a. Send Neighbor_Check_Response frame on port 1
67	FAULT_STATE (Backup)	Neighbor_Check_Request frame received on port 2	a. Send Neighbor_Check_Response frame on port 2
68	FAULT_STATE (Backup)	Neighbor_Check_Response frame received on port 1	a. Stop neighbor check timeout timer for port 1 and save neighbor status for port 1
69	FAULT_STATE (Backup)	Neighbor_Check_Response frame received on port 2	a. Stop neighbor check timeout timer for port 2 and save neighbor status for port 2

Event No.	Current State	Event	Action(s)
70	FAULT_STATE (Backup)	Neighbor check timer timeout on port 1	a. If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 1 and start neighbor check timeout timer for port 1 b. Else, send Neighbor_Status frame to active ring supervisor
71	FAULT_STATE (Backup)	Neighbor check timer timeout on port 2	a. If number of retries have not exceeded maximum limit, send Neighbor_Check_Request on port 2 and start neighbor check timeout timer for port 2 b. Else, send Neighbor_Status frame to active ring supervisor
72	FAULT_STATE (Backup)	Sign_On frame or Link_Status frame or Neighbor_Status frame received on port 1 or port 2 or Verify_Fault_Location service request received	a. Ignore
73	NORMAL_STATE (Backup)	Link lost on port 1	a. Send Link Status frame to active ring supervisor b. Stop beacon timeout timer for port 1 c. Set LastBcnRcvPort to 2, transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
74	NORMAL_STATE (Backup)	Link lost on port 2	a. Send Link Status frame to active ring supervisor b. Stop beacon timeout timer for port 2 c. Set LastBcnRcvPort to 2, transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
75	NORMAL_STATE (Backup)	Beacon frame from active ring supervisor with RING_FAULT_STATE received on port 1	a. If the ring state of previous beacon frame received on port 1 was not RING_NORMAL_STATE, return b. Else, stop beacon timeout timer for port 2 c. Set LastBcnRcvPort to 1, transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
76	NORMAL_STATE (Backup)	Beacon frame from active ring supervisor with RING_FAULT_STATE received on port 2	a. If the ring state of previous beacon frame received on port 2 was not RING_NORMAL_STATE, return b. Else, stop beacon timeout timer for port 1 c. Set LastBcnRcvPort to 2, transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
77	NORMAL_STATE (Backup)	Beacon timeout timer expired for port 1	a. Set LastBcnRcvPort to 2, transition to FAULT_STATE (Backup) and flush unicast MAC address learning table

Event No.	Current State	Event	Action(s)
78	NORMAL_STATE (Backup)	Beacon timeout timer expired for port 2	a. Set LastBcnRcvPort to 1, transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
79	NORMAL_STATE (Backup)	Beacon frame from active ring supervisor received on port 1	a. Restart port 1 beacon timeout timer
80	NORMAL_STATE (Backup)	Beacon frame from active ring supervisor received on port 2	a. Restart port 2 beacon timeout timer
81	NORMAL_STATE (Backup)	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 1	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return b. Else, stop beacon timeout timers for both ports c. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration d. Set LastBcnRcvPort to 1 e. Start beacon timeout timer for port 1 f. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
82	NORMAL_STATE (Backup)	Beacon frame from different ring supervisor MAC address than active ring supervisor is received on port 2	a. If new supervisor has lower precedence (or numerically lower MAC address in case of tie) than active supervisor, drop new Beacon frame and return b. Else, stop beacon timeout timers for both ports c. Save new active supervisor precedence, MAC address, VLAN ID and beacon timeout duration d. Set LastBcnRcvPort to 2 e. Start beacon timeout timer for port 2 f. Transition to FAULT_STATE (Backup) and flush unicast MAC address learning table
83	NORMAL_STATE (Backup)	Locate_Fault frame or Neighbor_Check_Request frame or Neighbor_Check_Response frame or Link_Status frame or Neighbor_Status frame received on port 1 or port 2 or Verify_Fault_Location service request received	a. Ignore
84	NORMAL_STATE (Backup)	Sign_On frame received on port 1	a. If first entry in participant list is self, drop frame and return b. Else, add node data to participant list and send frame on port 2

Event No.	Current State	Event	Action(s)
85	NORMAL_STATE (Backup)	Sign_On frame received on port 2	a. If first entry in participant list is self, drop frame and return b. Else, add node data to participant list and send frame on port 1

Notes:

- Network Topology attribute shall be updated at events 1, 2, 3, 53, 54, 59 and 61 to appropriate value.
- Network Status attribute shall be updated at events 1, 2, 3, 8, 11, 12, 13, 14, 37, 38, 39, 40, 41, 42, 43, 44, 53, 54, 55, 56, 57, 58, 59, 61, 73, 74, 75, 76, 77, 78, 81 and 82 to appropriate value.
- Ring Supervisor Status attribute shall be updated at events 1, 13, 14, 37, 38, 53, 54, 59 and 61 to appropriate value.
- Ring Faults Count attribute shall be updated at events 1, 13, 14, 37, 38, 39, 40, 41, 42, 43, 44, 53, 54, 59, 61 to appropriate value.
- Last Active Node 1 attribute shall be updated at events 1, 11, 12, 13, 14, 17, 21, 23, 26, 31, 41 and 43 to appropriate value.
- Last Active Node 2 attribute shall be updated at events 1, 11, 12, 13, 14, 18, 22, 24, 26, 32, 42 and 44 to appropriate value.
- Ring Protocol Partcipants Count attribute shall be updated at events 1, 13, 14, 37, 38 and 49 to appropriate value.
- Ring Protocol Partcipants List attribute shall be updated at events 1, 13, 14, 37, 38 and 49 to appropriate value.
- If in events 11 and 12, Beacon frames are received continuously for more than 3 consecutive beacon timeout durations on same port without the other port receiving any Beacon frames, then that constitutes a partial network fault condition. When such a condition is detected, the active ring supervisor shall block all traffic on port 2 resulting in network segmentation and update network status attribute. This condition requires user intervention to resolve. The location of fault can be identified through Verify_Fault_Location service.
- If in events 39, 40, 41, 42, 43, 44 the supervisor detects that 5 faults have occurred in a 30 second period, the supervisor shall remain in FAULT_STATE and block all traffic on port 2. Such a condition must be explicitly cleared by the user via the Clear_Rapid_Faults service of the DLR Object.

9-5.10 Performance Analysis

The example performance calculations below uses following key parameters/assumptions:

Variable	Description
NumNodes	Number of nodes in ring network, 50 for example.
BcnFrmDly	Delay due to a beacon frame in store and forward switching, 7 microseconds (64 byte frame with on wire overhead) for example.
AvgEipFrmDly	Delay due to average EtherNet/IP frame, 12 microseconds (128 byte frame with on wire overhead) for example.
MaxFrmDly	Delay due to maximum size Ethernet frame, 124 microseconds (1522 byte frame with on wire overhead) for example.
MaxDlyNodePrcnt	Percentage of number of nodes in ring on which beacon will be delayed by maximum size Ethernet frame, 10% for example.
IntSwtchDly	Internal switching delay on every node, 5 microseconds for example.
WirePrpgtDly	Signal propagation delay on 100 meters of copper media, 1 microsecond for example.
NodeProcDly	Processing delay on nodes for responding to ring frames and events, 25 microseconds for example.
BcnIntrvl	Beacon interval, should be less than half of the fastest connection RPI, 400 microseconds for example.

In order to provide predictable performance for network fault detection and reconfiguration all nodes directly on ring must dedicate highest priority queue to ring protocol frames and must implement strict priority scheduling for highest priority queue. With such a configuration, a ring protocol frame will encounter at most one lower priority frame delay on each node. For performance analysis, assume that all links on network operate at 100Mbps speed and in full duplex mode.

Ring Beacon frames are 64 bytes long including FCS and have an on wire overhead of 20 bytes of which 8 bytes are for preamble and start of frame delimiter pattern and 12 bytes are for inter frame gap. Ring Beacon frames with 20 byte on wire overhead take approximately BcnFrmDly = 7 microseconds on wire.

Assume that ring Beacon frames will be delayed on most nodes by a lower priority frame with an average size of 128 bytes including FCS. In a network with mostly EtherNet/IP traffic, on some nodes ring Beacon frames may not be delayed at all, in some other nodes they may be delayed by 256 byte frames and in some others it may be delayed by frames between these two extremes, for an average of 128 bytes. A 128 byte frame with 20 byte on wire overhead takes approximately AvgEipFrmDly = 12 microseconds on wire.

Assume that the nodes use store and forward switching architecture and that each node has an average internal switching overhead delay of IntSwtchDly = 5 microseconds. Assume propagation delay for copper media of 100 meters to be WirePrpgtDly = 1 microsecond. The total typical delay per node for ring Beacon frames is therefore

TypclDlyPerNode = BcnFrmDly + AvgEipFrmDly + IntSwtchDly + WirePrpgtDly

TypclDlyPerNode = 7 + 12 + 5 + 1 = 25 microseconds.

Assume that the ring Beacon frames would also be delayed on MaxDlyNodePrcnt = 10% of nodes by maximum sized Ethernet frames of 1522 bytes each or some combination of large frames on more than 10% of nodes that is equal to 10% of nodes with maximum sized frames. Such frames may be present on network for any reason including configuration, HMI, web etc. A 1522 byte frame with 20 byte on wire overhead takes approximately MaxFrmDly = 124 microseconds on wire. The total maximum delay per node for ring Beacon frames on these nodes is therefore

MaxDlyPerNode = BcnFrmDly + MaxFrmDly + IntSwtchDly + WirePrpgtDly

MaxDlyPerNode = 7 + 124 + 5 + 1 = 137 microseconds.

For a ring network comprised of NumNodes = 50 nodes, total maximum round trip time for Beacon frames is therefore

MaxRndTripTime = (NumNodes * (1 – MaxDlyNodePrcnt) * TypclDlyPerNode) +

(NumNodes * MaxDlyNodePrcnt * MaxDlyPerNode)

MaxRndTripTime = (50 * (1 – 0.1) * 25) + (50 * 0.1 * 137) = 1810 microseconds

For same network, minimum delay per node is when Beacon frame is not delayed by any other frame and therefore

MinDlyPerNode = BcnFrmDly + IntSwtchDly + PrpgtDly

MinDlyPerNode = 7 + 5 + 1 = 13 microseconds

For a ring network comprised of NumNodes = 50 nodes, total minimum round trip time is therefore

MinRndTripTime = NumNodes * MinDlyPerNode

MinRndTripTime = 13 * 50 = 650 microseconds.

In general, lower beacon interval provides faster ring recovery performance. Beacon interval should be less than half of the fastest connection RPI in the network to prevent connection timeouts. Assume a beacon interval of BcnIntrvl = 400 microseconds which constitutes 1.75% of network bandwidth and is suitable for high performance CIP motion connections with 1 millisecond RPI and also works for slower I/O connections.

For choosing beacon timeout, consider a first Beacon frame transmitted at time Tx_1 facing minimum round trip time delay and arriving at active supervisor ports at time $Rx_1 = Tx_1 +$ MinRndTripTime $= Tx_1 + 650$ microseconds. Assume that a second Beacon frame transmitted at time $Tx_2 = Tx_1 +$ BcnIntrvl $= Tx_1 + 400$ microseconds is lost on route due to frame corruption. A third Beacon frame transmitted at time $Tx_3 = Tx_2 +$ BcnIntrvl $= Tx_2 + 400$ microseconds facing maximum roundtrip delay will arrive at active supervisor ports at time $Rx_3 = Tx_3 +$ MaxRndTripTime $= Tx_3 + 1810$ microseconds. The maximum arrival delay of third Beacon frame on active supervisor ports after first Beacon frame is therefore equal to $Rx_3 - Rx_1 = (Tx_3 + 1810) - (Tx_1 + 650) = (Tx_1 + 400 + 400 + 1810) - (Tx_1 + 650) = 1960$ microseconds. Hence the beacon timeout duration should be equal to BcnTimeout = 1960 microseconds.

Assume end node processing delay of NodeProcDly = 25 microseconds for responding to ring frames or events.

For the network described, following will be the worst case performance for ring nodes that rely on Beacon frame mechanism:

1. Faults that are detectable in physical layer: This is the most common type of faults. The worst case scenario happens when the fault occurs half way across ring network from active ring supervisor. It will take half round trip time for Link Status frames to reach from neighboring nodes of fault to active ring supervisor. It will take another half round trip time for Beacon frame with FAULT_STATE from active supervisor to reach farthest node or for ring nodes to timeout Beacon frames whichever happens earlier. End node processing delay is involved for three times, once at fault neighbor, once at supervisor and once at farthest node. The total worst case delay is therefore

   PhysclLyrFltDlyBcn = (2 * 0.5 * MaxRndTripTime) + (3 * NodeProcDly)

   PhysclLyrFltDlyBcn = 1810 + (3 * 25) = 1885 microseconds.

2. Faults that are not detectable in physical layer: This type of faults is relatively rare. The worst case scenario happens when the fault occurs half way across ring network from active ring supervisor. It will take half round trip time for last Beacon frame from near fault location to reach active ring supervisor. It will take another beacon timeout period for active supervisor and ring nodes to timeout Beacon frames. End node processing delay is involved once at all nodes. The total worst case delay is therefore

   NonPhysclLyrFltDlyBcn = (0.5 * MaxRndTripTime) + BcnTimeout + NodeProcDly

   NonPhysclLyrFltDlyBcn = (0.5 * 1810) + 1960 + 25 = 2890 microseconds.

3. Network restoration to normal mode of operation: It is equal to a total of one beacon interval for a beacon to be generated, one maximum round trip time for beacon and another half maximum round trip time for Beacon frame with RING_NORMAL_STATE to reach farthest node. End node processing delay is involved twice, once at ring supervisor and once at all nodes. The total worst case delay is therefore

   RingRstrDlyBcn = BcnIntrvl + (1.5 * MaxRndTripTime) + (2 * NodeProcDly)

   RingRstrDlyBcn = 400 + (1.5 * 1810) + (2 * 25) = 3165 microseconds.

For the network described, following will be the worst case performance for ring nodes that rely on Announce frame mechanism:

1. Faults that are detectable in physical layer: This is the most common type of faults. The worst case scenario happens when the fault occurs half way across ring network from active ring supervisor. It will take half round trip time for Link Status frames to reach from neighboring nodes of fault to active ring supervisor. It will take another half round trip time for Announce frame with FAULT_STATE from active ring supervisor to reach farthest node. End node processing delay is involved for three times, once at fault neighbor, once at supervisor and once at farthest node. The total worst case delay is therefore

   PhysclLyrFltDlyAnnc = (2 * 0.5 * MaxRndTripTime) + (3 * NodeProcDly)

   PhysclLyrFltDlyAnnc = 1810 + (3 * 25) = 1885 microseconds.

2. Faults that are not detectable in physical layer: This type of faults is relatively rare. The worst case scenario happens when the fault occurs half way across ring network from active ring supervisor. It will take half round trip time for last Beacon frame from near fault location to reach active ring supervisor. It will take another beacon timeout period for active supervisor to timeout Beacon frames. It will take another half round trip time for Announce frame with FAULT_STATE from active supervisor to reach farthest node. End node processing delay is involved twice, once at supervisor and once at end node. The total worst case delay is therefore

   NonPhysclLyrFltDlyAnnc = (2 * 0.5 * MaxRndTripTime) + BcnTimeout + (2 * NodeProcDly)

   NonPhysclLyrFltDlyAnnc = 1810 + 1960 + (2 * 25) = 3820 microseconds.

3. Network restoration to normal mode of operation: It is equal to a total of one beacon interval for a beacon to be generated, one maximum round trip time for beacon and another maximum round trip time for Announce frame to reach farthest node. End node processing delay is involved twice, once at supervisor and once at end node. Note that this is the total delay for an Announce based node to flush its unicast MAC table. The ring would have been restored earlier per calculation for Beacon based nodes. The total worst case delay is therefore

   RingRstrDlyAnnc = BcnIntrvl + (2 * MaxRndTripTime) + (2 * NodeProcDly)

   RingRstrDlyAnnc = 400 + (2 * 1810) + (2 * 25) = 4070 microseconds.

Based on similar calculations Table 9-5.20 below provides configuration parameters and worst case performance numbers for different ring network node numbers. It should be noted that though these performance numbers will work for most cases, deviation from assumptions outlined above will require recalculation of numbers based on procedure described above. For example, if any link in ring is set to operate at 10Mbps speed and/or half duplex mode the numbers must be adjusted (for 10Mbps, multiply by 10).

Table 9-5.20 Example Ring Configuration Parameters and Performance

Number of Ring Nodes	Beacon Interval (usecs)	Round Trip Time[1] (usecs)	Beacon Timeout (usecs)	Physical Layer Faults Recovery Delay 1 (usecs)	Non-physical Layer Faults Recovery Delay for Beacon Frame Based Nodes (usecs)	Non-physical Layer Faults Recovery Delay for Announce Frame Based Nodes (usecs)	Ring Restore Delay for Beacon frame Based Nodes (usecs)	Ring Restore Delay for Announce frame Based Nodes (usecs)
25	400	905	1380	980	1858	2335	1808	2260
50 (nominal network size)	400	1810	1960	1885	2890	3820	3165	4070
100	400	3620	3120	3695	4955	6790	5880	7690
150	400	5430	4280	5505	7020	9760	8595	11310
200	400	7240	5440	7315	9085	12730	11310	14930
250	400	9050	6600	9125	11150	15700	14025	18550

1 Same for Beacon and Announce frames based nodes.

Important: When non-DLR nodes are present in the ring, recovery and restoration delays are as they are for Announce-based nodes, provided the non-DLR nodes follow the requirements specified in section 9-5.5. If non-DLR nodes don't follow the requirements, recovery and restoration delays are unpredictable.

When using non-DLR switches in the ring, unicast packet loss may occur, as described in section 9-5.7.3, Non-DLR Switches.

Volume 2: EtherNet/IP Adaptation of CIP

Chapter 10: Bridging & Routing

Contents

10-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the definition of CIP bridging and routing that are EtherNet/IP specific. At this time, no such additions exist.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Appendix A: Explicit Messaging Services

Contents

A-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the definition of CIP explicit messaging services that are EtherNet/IP specific. At this time there are no such additions.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Appendix B: Status Codes

Contents

B-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the definition of CIP error codes that are EtherNet/IP specific. At this time there are no such additions.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Appendix C: Data Management

Contents

C-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the CIP Data Management specification that are EtherNet/IP specific. At this time there are no such additions.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Appendix D: Engineering Units

Contents

D-1 Introduction

This chapter of the EtherNet/IP specification contains additions to the list of CIP engineering units that are EtherNet/IP specific. At this time, there are no such additions.

This page is intentionally left blank

Volume 2: EtherNet/IP Adaptation of CIP

Appendix E: EtherNet/IP Quick Connect

Contents

E-1 Introduction

In many automotive applications, robots, tool changers and framers are required to quickly exchange tooling fixtures which contain a section or segment of an industrial network. This requires the network and nodes to be capable of quickly connecting and disconnecting, both mechanically, and logically.

While the mechanical means for connecting and disconnecting tooling exists, achieving a quick re-establishment of a logical network connection between a network controller and a fully powered-down node on Ethernet can take as much as 10 or more seconds. This is too slow for applications that require very short cycle times.

The time in which a robot arm first makes electrical contact with a new tool, until the mechanical lock being made, is typically 1 second. In applications where the tools are constantly being connected and disconnected, the nodes need to be able to achieve a logical connection to the controller and test the position of the tool in less than 1 second from the time the tool and the robot make an electrical connection. This means that the node needs to be able to power up and establish a connection in approximately 500 ms. This section discusses the requirements and optimizations necessary for the controller and the Quick Connect target node that may affect the overall logical connection time on an EtherNet/IP network when physically connecting and disconnecting.

It should be noted that controller and robotic application behavior is outside the scope of this specification.

The Quick Connect feature is an option enabled on a node-by-node basis. When enabled, the Quick Connect feature will direct EtherNet/IP target devices to quickly power up and join an EtherNet/IP network.

In order for Quick Connect devices to power up as quickly as possible, the following list of recommendations for Quick Connect device manufacturers should be followed.

- Minimize the hardware delay at power-up and reset as much as possible.
- Design devices with embedded switches and (at least) 2 external Ethernet ports, so that devices may be deployed in a linear topology to eliminate the start up time of a separate switch on the tool.
- Optimize power up self-test routines and LED tests.

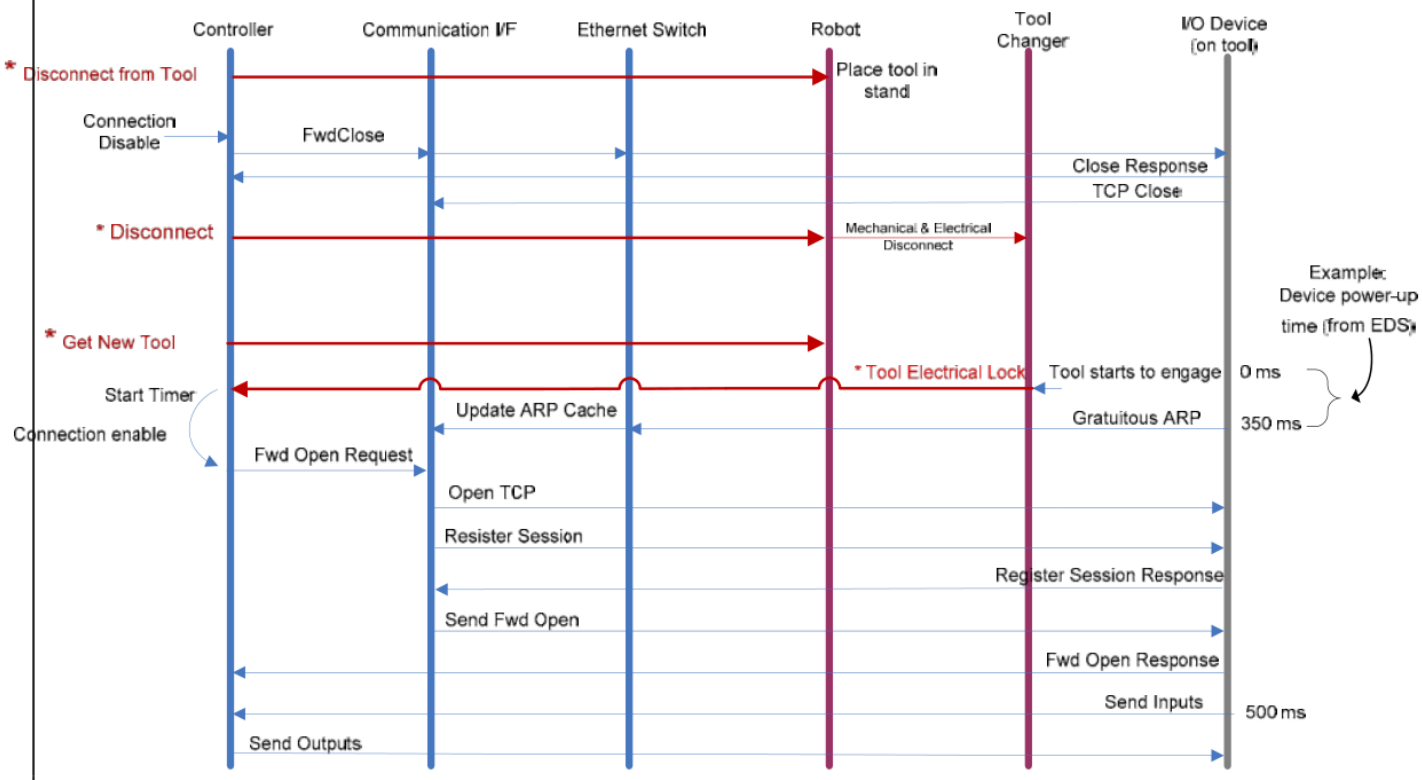
The Quick Connect feature is enabled within a target device through the non-volatile EtherNet/IP Quick Connect attribute (12) in the TCP/IP object. A device shall have this feature disabled as the factory default. See section 5-3.3.2.11.

The goal for Quick Connect connection time is 500ms. Specifically, this is defined as the guaranteed repeatable time between the electrical contact of power and Ethernet signals at the tool changer, and when the newly connected devices are ready to send the first CIP I/O data packet.

Quick Connect connection time is comprised of several key time durations. The majority of the Quick Connect connection time is due to the Quick Connect target devices' power-up time. Also contributing to the connection time is the amount of time it takes a controller to detect the newly attached device and send a Forward Open to start the connection process. The overall 500ms Quick Connect connection time is additive, and consists of the Quick Connect devices' power-up time, the controller's connection establishment time, and actual network communication time. Also, the network communication time is dependent on the network topology. For instance, in a linear topology, the network communication time will be dependent on all devices powering up, plus the delay through all of the devices. The final application connection time assumes that connections to ALL of the I/O devices on the tool have been established.

The following figure shows the events, states, and sequence in which a controller shall discontinue communications with a device on a given tool and then establish a connection to a device on a new tool. Note: There can be multiple I/O devices on the tool. This sequence is repeated for each connection from the controller to the I/O devices on the tool.

Figure E-1.1 Quick Connect System Sequence Diagram (application behavior steps are in red)



In Figure E-1.1, actions marked by * are the typical application actions, but may vary and are outside the scope of the spec. Refer to section E-3 for specific requirements and actions of the controller.

There shall be two classes of Quick Connect devices.

- Class A Quick Connect target devices shall be able to power-up, send the first Gratuitous ARP packet, and be ready to accept a TCP connection in <= 350ms.
- Class B Quick Connect target devices shall be able to power-up, send the first Gratuitous ARP packet, and be ready to accept a TCP connection in <= 2 seconds.

E-2 EtherNet/IP Target Requirements

EtherNet/IP target devices supporting Quick Connect shall adhere to the following requirements.

- Power-up diagnostics and LED tests may be delayed as needed in order to meet the desired startup time. Diagnostics may be performed in the background after startup
- Quick Connect devices shall implement the ability to set forced speed/duplex mode (for 10/100 Mb Ethernet at least), via the Ethernet Link Object. It is recommended that users set forced speed and duplex mode on all of a device's ports during Quick Connect device commissioning in order to meet the fastest Quick Connect timing requirements.
- When in Quick Connect mode, a port configured for forced speed and duplex mode, Quick Connect devices shall not use Auto-MDIX (detection of the required cable connection type). This detection may take more time than the allotted Quick Connect system connection time. To enable the use of straight-thru cables when Auto-MIDX is disabled, the following rules shall be applied:
  1. On a device with only one port: the port shall be configured as MDI.
  2. On devices with 2 external Ethernet ports:
     a. The labels for the 2 external ports shall include an ordinal indication (e.g.: Port 1 and Port 2, or A and B).
     b. The port with the lower ordinal indication shall be configured as MDI.
     c. The port with the upper ordinal indication shall be configured as MDIX.

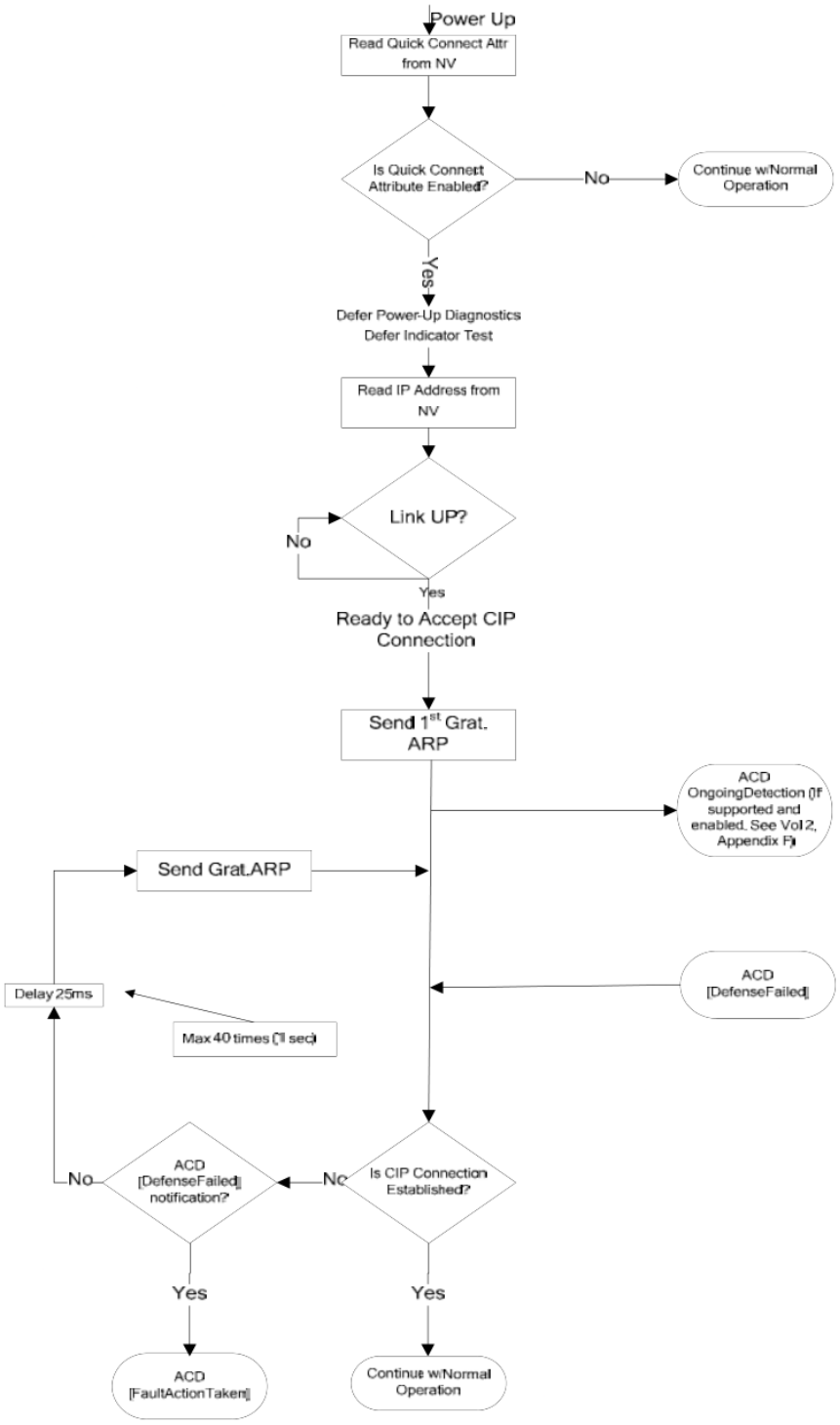
  Note: For DLR capable devices this requirement overrides the requirement that DLR capable devices shall have "forced Auto-MDIX" when speed & duplex are fixed.
- The target device shall support EtherNet/IP Quick_Connect attribute (12) in the TCP/IP Object that enables the Quick Connect feature. See section 5-3.3.2.
- The target device shall have the Quick Connect keywords and values included in the device's EDS file. See section 7-6.
- When a Quick Connect target's I/O connection is closed via a Forward Close, the target shall close the TCP connection after sending the Forward Close Response. It is important to clean up resources before powering down so that a new connection can be quickly established with the replacement node by the controller. Having the target device close the TCP connection, as opposed to the originator, forces a timelier cleanup of resources in the originator needed for the next quick connection establishment. In this scenario, the EtherNet/IP encapsulation session is also cleaned up.
- When the target device is ready to join the network, it shall issue a gratuitous ARP which will cause any needed update to the ARP table of the controller for that IP address. Because ARP is not a guaranteed service it may be necessary to send multiple gratuitous ARP messages to ensure that the controller's ARP cache is updated. The device shall continue to issue the gratuitous ARP every 25ms for a maximum of 40 times (1 second), until an I/O connection is established.

- When the Quick Connect feature is enabled, if IPv4 Address Conflict Detection (see Volume 2, Appendix F) is supported and enabled, the device shall skip the initial Address Probing and Address Announcement phases and enter the Ongoing Conflict Detection phase immediately after putting the IP address in use (refer to Figure E-2.1 and Figure F-1.1). All other aspects of IPv4 Address Conflict Detection behavior shall be as specified in Volume 2, Appendix F. The ACD state machine shall run independent of the Quick Connect startup behavior shown in Figure E-2.1.

Important: As a consequence of skipping initial address probing, a Quick Connect device whose IP address conflicts with an existing device may disrupt communications to the existing device. The user must assume responsibility for ensuring that no nodes exist with the same IP address and that no more than one connection originator is configured to access the same Quick Connect device.

Figure E-2.1 shows the typical target device power-up logic as it pertains to Quick Connect. Note that interaction with IPv4 Address Conflict Detection is applicable only if the device supports IPv4ACD.

Figure E-2.1 Typical Target Device Power Up Logic for Quick Connect



E-3 Controller Requirements

- An EtherNet/IP controller device that participates in Quick Connect connection establishment shall have the Quick Connect keywords and values included in its EDS file. See Table 7-6.1
- In a Quick Connect system, the controller must have the capability to open and close connections to the Quick Connect target devices based on system events. For example, two such system events might be:
  1. When the controller is done with a given tool (set of IP addresses on the tool)
  2. When the controller needs to enable connections to the new tool (set of IP addresses on the “new” tool) when it receives the electrical lock input signal from the tool changer indicating that the “new” tool was in place.

Volume 2: EtherNet/IP Adaptation of CIP

Appendix F: Address Conflict Detection

Contents

F-1 Introduction

Address conflict detection (ACD) is a mechanism that EtherNet/IP devices can use to detect and act upon IPv4 address conflicts. The ACD mechanism deployed in EtherNet/IP conforms to the IETF RFC 5227.

The requirements specified in RFC 5227 are included by reference except where superseded by normative statements herein. This section also specifies additional requirements for EtherNet/IP devices with respect to the IPv4 ACD mechanism.

- ACD is RECOMMENDED for EtherNet/IP devices.
- An EtherNet/IP device, implementing ACD, SHALL conform to the ACD mechanism specified in IETF RFC 5227 except where superseded by normative statements herein
- An EtherNet/IP device, executing ACD, SHALL execute the ACD mechanism regardless of the method used by the device for obtaining its IP parameter set.
- An EtherNet/IP device, executing ACD, and not executing the Quick Connect algorithm SHALL execute the ACD mechanism before using an IP parameter set.
- EtherNet/IP Devices SHALL respond to ARP Probes.

F-1.1 Overview of ACD

The IPv4 ACD mechanism described by RFC 5227 involves the following activities:

- Initial Address Probing & Conflict Detection: Before using an IP address the device issues ARP Probes to detect whether the address is in use by another device.
- Address Announcement: An ARP Request packet is sent after determination that there are no IP address conflicts.
- Ongoing Conflict Detection: Ongoing process that is in effect for as long as a device is using an IP Address.
- AddressDefense: Procedure used to resolve an address conflict.

The following sections specify additional requirements for EtherNet/IP devices that implement the ACD mechanism.

F-1.2 ACD Behavior

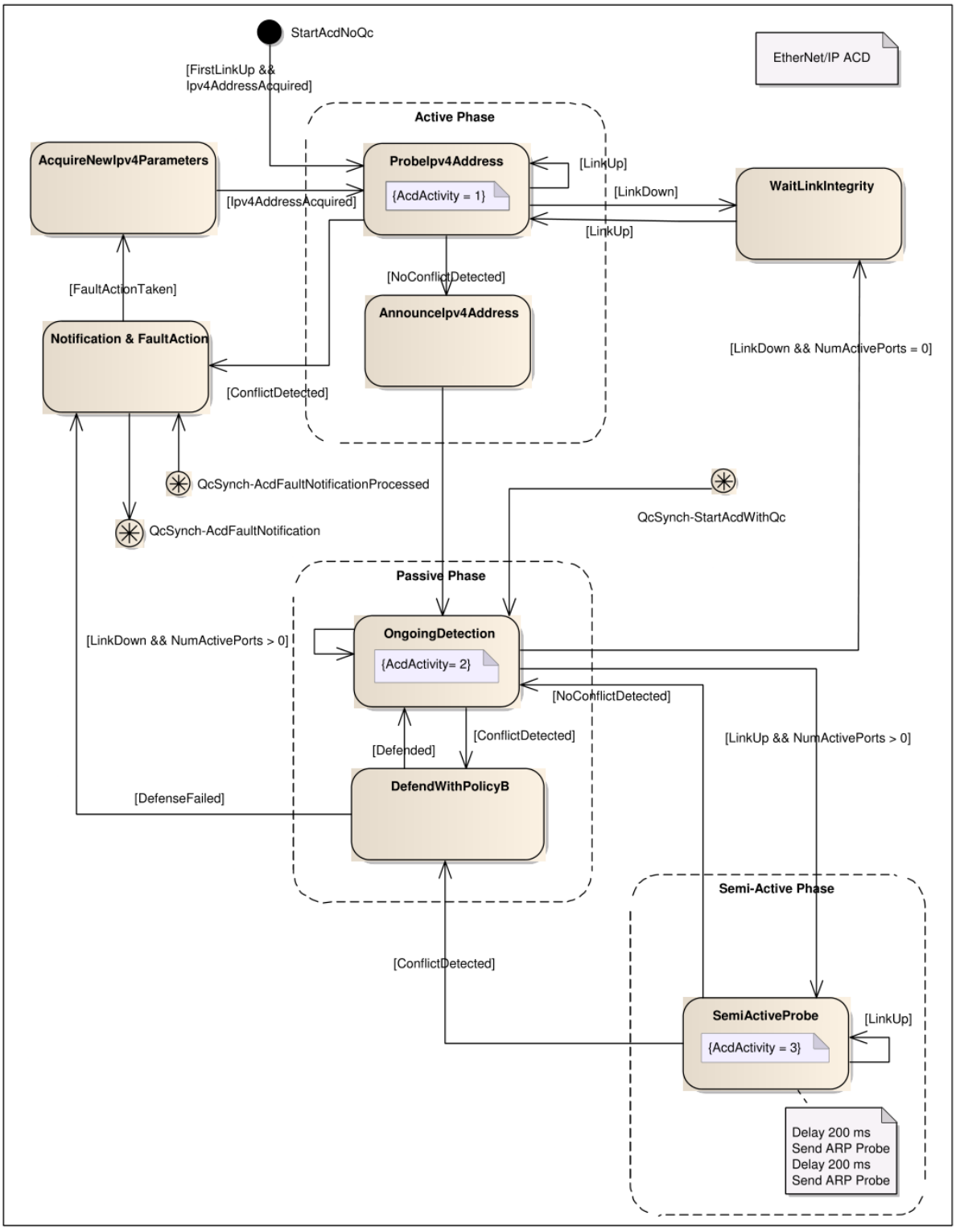
The principles of operation for the ACD mechanism are specified in IETF RFC 5227. Of particular note is the RFC 5227 requirement from section 2.1 that "… a host implementing this specification MUST test to see if the address is already in use … when a link-state change signals that an Ethernet cable has been connected …"

In addition, this CIP volume further refines the ACD behavior description in the following ways:

- The activities of ACD are grouped into regions, namely
  - Active Phase,
  - Passive Phase and
  - Semi-Active Phase.
- A more complete set of link_up transitions are included in the diagram.
- Both single port and multi-port devices are accommodated.

An EtherNet/IP device SHALL implement the ACD mechanism with the behavior specified in Figure F-1.1.

Figure F-1.1 ACD Behavior



F-1.2.1 IPv4 ACD Timing Constants

Section 1.1 of RFC 5227 specifies a number of timing-related constants. Some of these constants are not optimal for EtherNet/IP applications and networks. In particular, the start-up delays caused by the ARP Probe intervals would be unacceptable in the industrial setting. Adaptation of the specified alternate values is consistent with section 1.3 of RFC 5227 on Applicability.

EtherNet/IP devices that provide ACD SHALL implement the following alternate values for the parameters defined in section 1.1 of RFC 5227.

- PROBE_WAIT (180..220) ms (initial random delay)
- PROBE_NUM 4 (number of probe packets)
- PROBE_MIN 180 ms (minimum delay until repeated probe)
- PROBE_MAX 220 ms (maximum delay until repeated probe)
- ANNOUNCE_WAIT (180..220) ms (delay before announcing)

F-1.2.2 ProbeIpv4Address

See sections 2.1 and 2.1.1 of IETF RFC 5227.

F-1.2.3 AnnounceIpv4Address

See section 2.3 of IETF RFC 5227

F-1.2.4 ConfigStack

Once the IPv4 Address has been successfully probed and announced, the TCP/IP stack is configured to use this IPv4 address and its associated parameters such as subnet mask, etc.

Once the stack is configured with the validated IP Parameter set, the OngoingDetection activity is initiated.

F-1.2.5 WaitLinkIntegrity

The WaitLinkIntegrity activity waits for the occurrence of a LinkUp event from an Ethernet port.

This activity is initiated in two scenarios.

- One is when a single port device experiences a LinkDown event.
- The other is when a multi-port device experiences a LinkDown event and all of its other Ethernet ports are also down.

When a LinkUp event occurs the ProbeIpv4Address activity is then initiated.

When a device detects that Ethernet link integrity has been lost and then regained (e.g., the network cable has been removed and replaced), the device SHALL restart the initial ProbeIpv4Address activity.

F-1.2.6 Notification & Fault Action

See section 1 of IETF RFC 5227.

In addition to RFC 5227 behavior, when an address conflict is detected, EtherNet/IP devices SHALL take the following fault actions:

- Set the device state to Recoverable Fault (Module Status LED flashing red), and
- Set the Network Status LED to solid red.

There are 2 synchronization transitions between the ACD Notification & FaultAction activity and the Quick Connect activity diagram shown in Appendix E, Figure E-2.1.

The QcSynch-AcdFaultNotification is a synchronization transition to the Quick Connect behavior diagram that occurs either when the ACD mechanism has a ConflictDetected fault from the ProbeIpAddress acticity or a DefenseFailed fault from the DefendWithPolicyB activity.

The QcSynch-AcdFaultNotificationProcessed is a synchronization transition from the Quick Connect behavior diagram to the Notification & FaultAction activity that occurs after Quick Connect has processed the previous AcdFaultNotification.

F-1.2.7 AcquireNewIpv4Parameters

See section 1 of IETF RFC 5227.

After notification of the detection of an IPv4 Address conflict, the device may be designed to obtain or use an alternate IPv4 address. The design of this method and its resulting selection of IPv4 Parameters are vendor specific.

F-1.2.8 OngoingDetection

See section 2.4 of IETF RFC 5227.

The QcSynch-StartAcdWithQc is a synchronization transition with the Quick Connect behavior diagram shown in Appendix E, Figure E-2.1. If Quick Connect is enabled in the device, the activity for ACD begins at this point.

F-1.2.9 DefendWithPolicyB

See section 2.4 of IETF RFC 5227.

> "Address Conflict Detection is not limited to only the time of initial interface configuration, when a host is sending ARP Probes. Address Conflict Detection is an ongoing process that is in effect for as long as a host is using an address." [RFC 5227 sec 2.4]

After an IP address has successfully been probed and put into use, a device SHALL perform ongoing conflict detection and defense according to RFC 5227.

If an address conflict is detected, the device SHALL defend its address per alternative (b) in RFC 5227 section 2.4.

If a conflict persists (as defined by RFC 5227) the device SHALL follow the behavior as when a conflict is detected during initial probing (device state, LEDs, DHCP behavior, etc.) and execute the specified notification & fault actions.

The ACD mechanism as specified in IETF RFC 5227 section 2.1 states that "A host must not perform this check periodically as a matter of course". Accordingly periodic ARP Probes must not be sent as part of ongoing detection. However periodic ARP Probes allow the module to detect conflicts with devices that may not have been connected to the network at the time of initial probing, or when switches have dropped the initial ARP Probes due to initial forwarding delay.

EtherNet/IP devices that support this ongoing periodic probe SHALL implement the following parameters and defined values which are not defined in IETF RFC 5227.

- ONGOING_PROBE_MIN 80 seconds (minimum interval for ongoing probes)
- ONGOING_PROBE_MAX 165 seconds (maximum interval for ongoing probes)

It is RECOMMENDED that the initial transmission delay for the first periodic ARP Probe have a pseudo-random value within the range of ONGOING_PROBE_MIN and ONGOING_PROBE_MAX. ("randomized" using the Ethernet MAC address as shown by example in **Error! Reference source not found.** below).

When performing ongoing detection it is RECOMMENDED that devices issue periodic ARP Probes within the range of ONGOING_PROBE_MIN and ONGOING_PROBE_MAX.

F-1.2.9.1 Example Pseudo-random Delay Algorithm

The following is an Xorshift Random Number Generator (RNG) algorithm that is based on the work of George Marsaglia from Florida State University [1].

This RNG is used to produce an initial pseudo-random delay before sending out the first periodic ARP probe

Assuming an initial PROBE_WAIT of 200 ms +/- 20 ms, gives a PROBE_WAIT with a range of (180 .. 220).

The strategy is to produce a random number in the range of PROBE_WAIT, ie (180 .. 200) This is achieved by calculating a random offset from the beginning of the range.

Accordingly, PROBE_WAIT is calculated according to the following formula,

PROBE_WAIT = PROBE_WAIT_MIN + Δrn

Δrn is the random offset ranging from (0 .. to (PROBE_WAIT_MAX – PROBE_WAIT_MIN) and calculated according to the following formula,

Δrn = (PROBE_WAIT_MAX – PROBE_WAIT_MIN) * R256 / 256.

R256 is a random number in the range (0 .. 255) calculated using the algorithm below.

[1] Xorshift RNGs, George Marsaglia, Florida State University, Journal of Statistical Software, Volume 8, Issue 14, http://www.jstatsoft.org/v08/i14/paper.

The RNG algorithm uses an IEEE 802.3 MAC address, eg 12-34-56-78-9A-BC, written here in canonical notation. Canonical notation represents the order in which the bytes of the MAC address are transmitted, left to right, on the wire; i.e. 0x12 sent first, 0x34 sent next, and so on. The MAC address is mapped to an array, EnetAddr[], of 6 bytes as follows, using the example MAC address .

```
EnetAddr[0] = 0x12;
EnetAddr[1] = 0x34;
EnetAddr[2] = 0x56;
EnetAddr[3] = 0x78;
EnetAddr[4] = 0x9A;
EnetAddr[5] = 0xBC;
```

R256 is then calculated using the following Xorshift RNG.

```
R256 = EnetAddr[0];
R256 = (R256 << 1 ) ^ EnetAddr[1];
R256 = (R256 << 1 ) ^ EnetAddr[2];
R256 = (R256 << 1 ) ^ EnetAddr[3];
R256 = (R256 << 1 ) ^ EnetAddr[4];
R256 = (R256 << 1 ) ^ EnetAddr[5];
R256 = R256 % 256;
```

Where << is the left shift operator, ^ is the exclusive OR operator, and % is the modulus operator.

F-1.2.10 SemiActiveProbe

The SemiActiveProbe activity is initiated when a multi-port device receives a LinkUp event while other ports are still active.

The SemiActiveProbe activity is a modified probing activity in which only two ARP probes are sent using probe delays of 200 ms.

The SemiActiveProbe activity is defined to reduce the amount of ARP broadcast traffic that will be initiated from LinkUp activity on multi-port devices.